O'REILLY®

TURING 图灵程序设计丛书



数据科学实战

DOING DATA SCIENCE

[美] Rachel Schutt　Cathy O'Neil 著
冯凌秉　王群锋 译

中国工信出版集团　人民邮电出版社
POSTS & TELECOM PRESS

数字版权声明

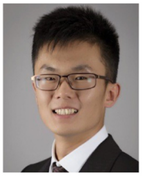
图灵社区的电子书没有采用专有客户端，您可以在任意设备上，用自己喜欢的浏览器和PDF阅读器进行阅读。

但您购买的电子书仅供您个人使用，未经授权，不得进行传播。

我们愿意相信读者具有这样的良知和觉悟，与我们共同保护知识产权。

如果购买者有侵权行为，我们可能对该用户实施包括但不限于关闭该帐号等维权措施，并可能追究法律责任。

译者介绍



冯凌秉

澳大利亚国立大学统计学博士，本科和研究生分别毕业于中南财经政法大学和中国人民大学。现在，他任职于江西财经大学金融管理国际研究院，任讲师、硕士生导师，研究方向为应用统计与金融计量。



王群锋

毕业于西安电子科技大学，现任职于IBM西安研发中心，从事下一代统计预测软件的开发运维工作。

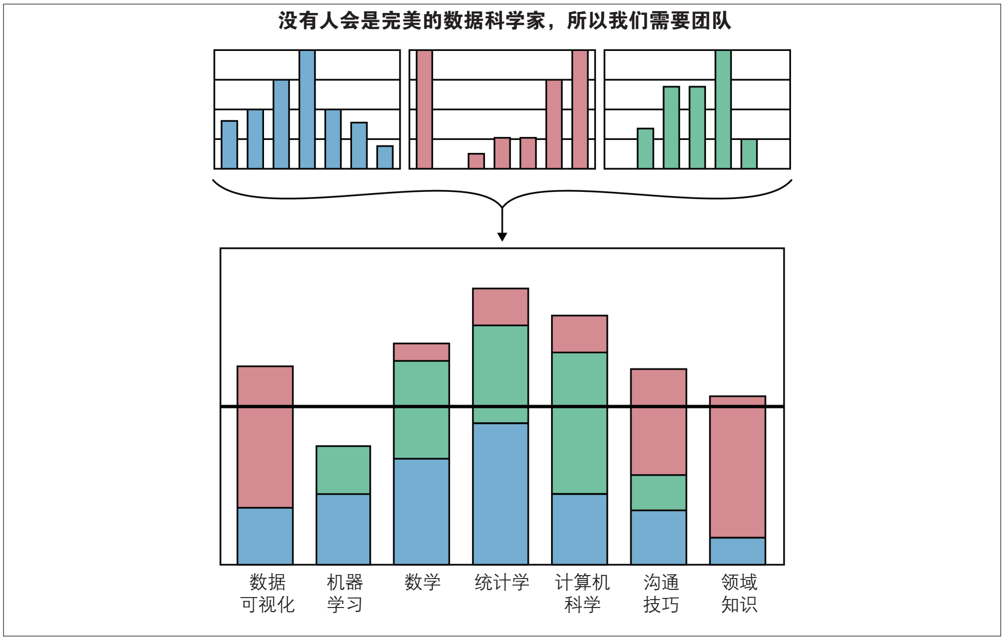


图 1-3：数据科学团队的知识结构由每个成员的知识结构叠加而来，在组建团队时，要让团队技能与所解决的问题大致匹配

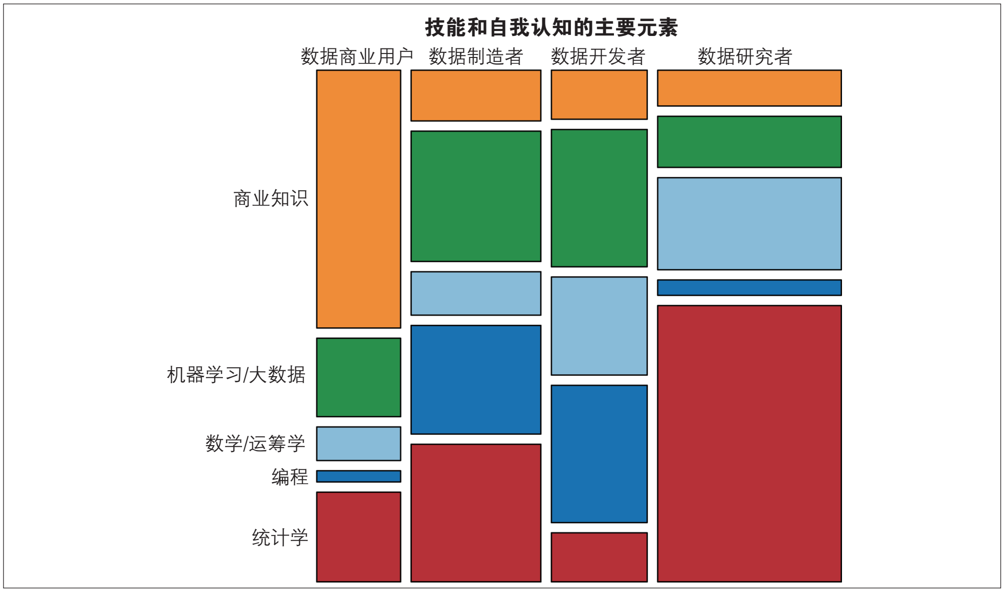


图 1-4：此图使用聚类算法描述了数据科学的子领域，源自 Harlan Harris、Sean Murphy 和 Marck Vaisman 基于 2012 年年中对数百名数据科学从业者的调查所著的 *Analyzing the Analyzers*（O'Reilly）

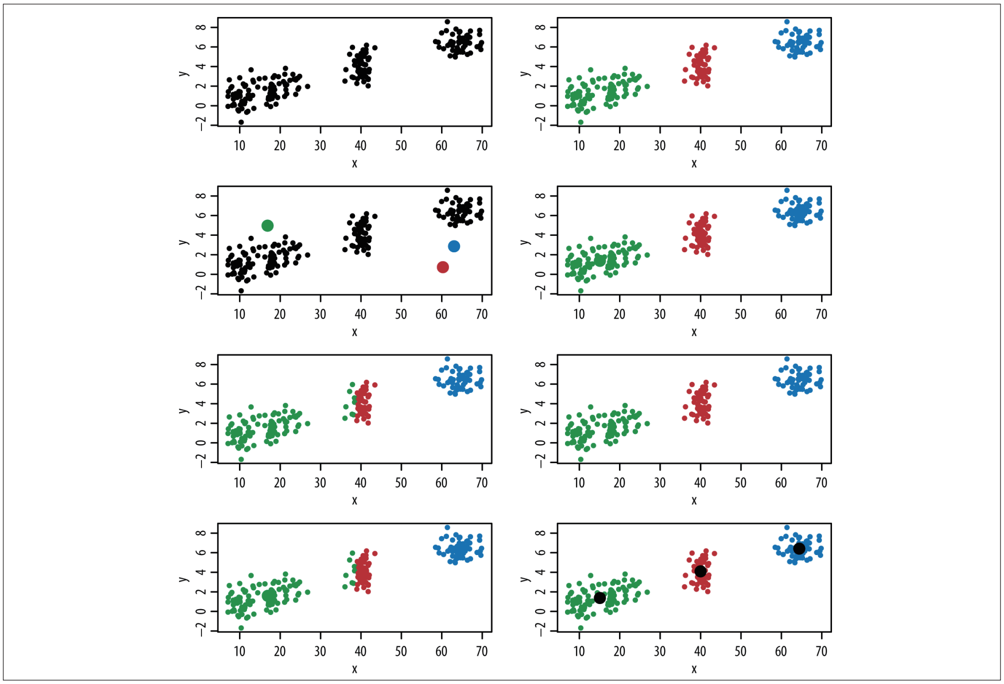


图 3-9：二维空间上的聚类过程，先看左半边从上往下，再看右半边从上往下

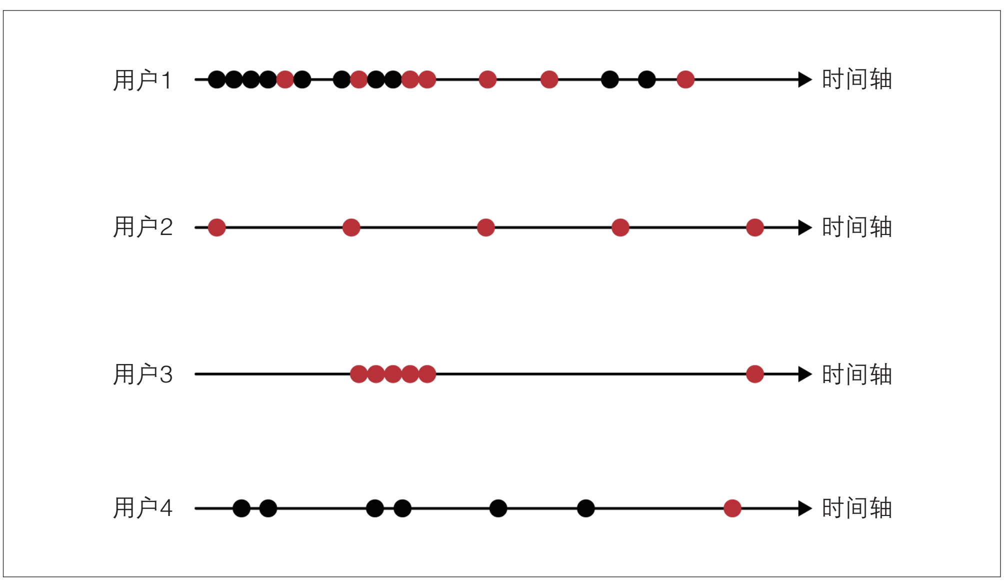


图 6-3：在用户的时间序列图中，用不用的颜色代表用户不同的动作类型。红色表示“点赞”，黑色表示“点衰”

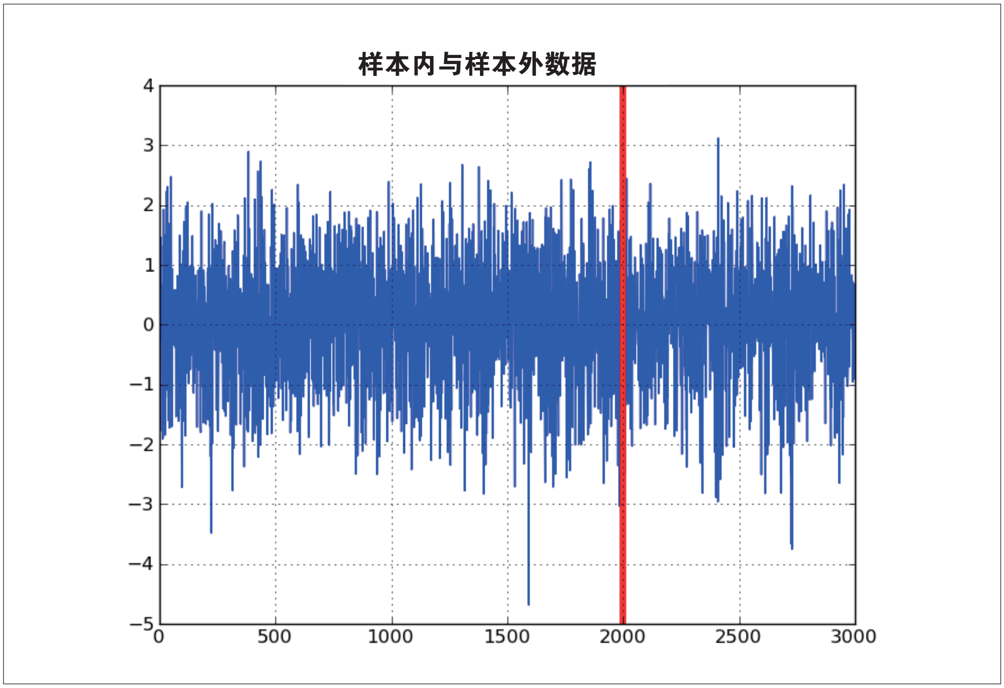


图 6-7：样本期内的数据永远发生在样本期外数据之前，红色线代表了模型建立的时点

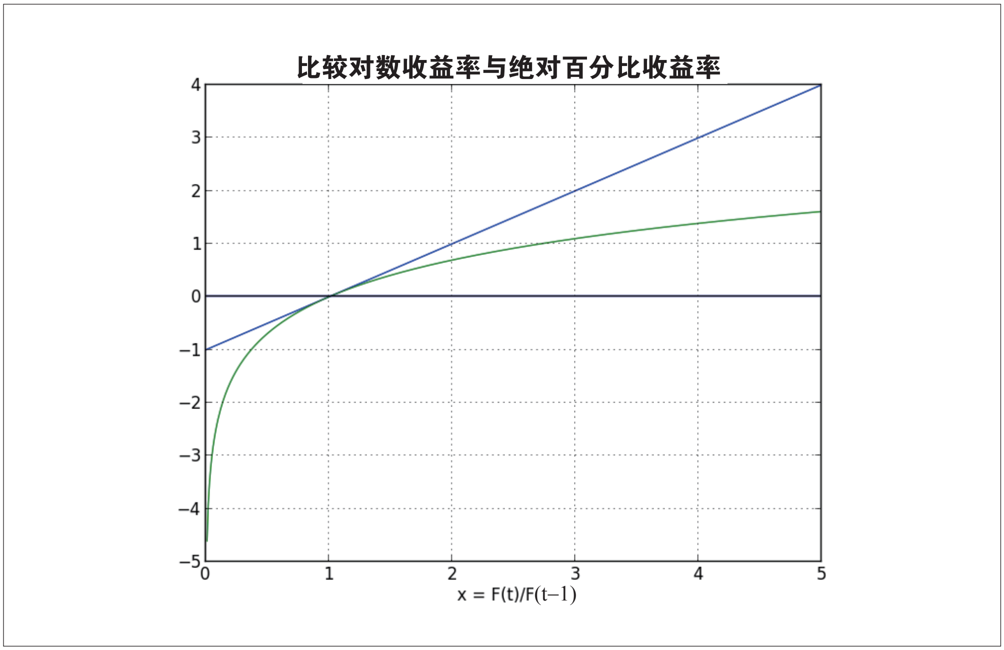


图 6-8：对数和绝对百分比收益率曲线对比图

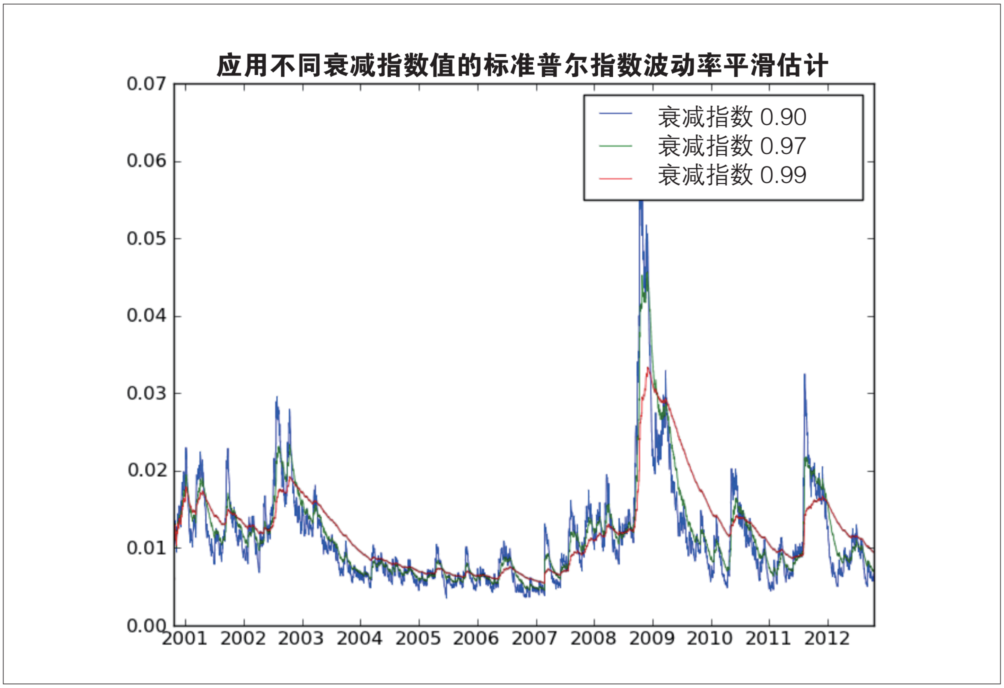


图 6-12：标准普尔指数的波动率的指数平滑估计：使用了三个不同大小的指数值

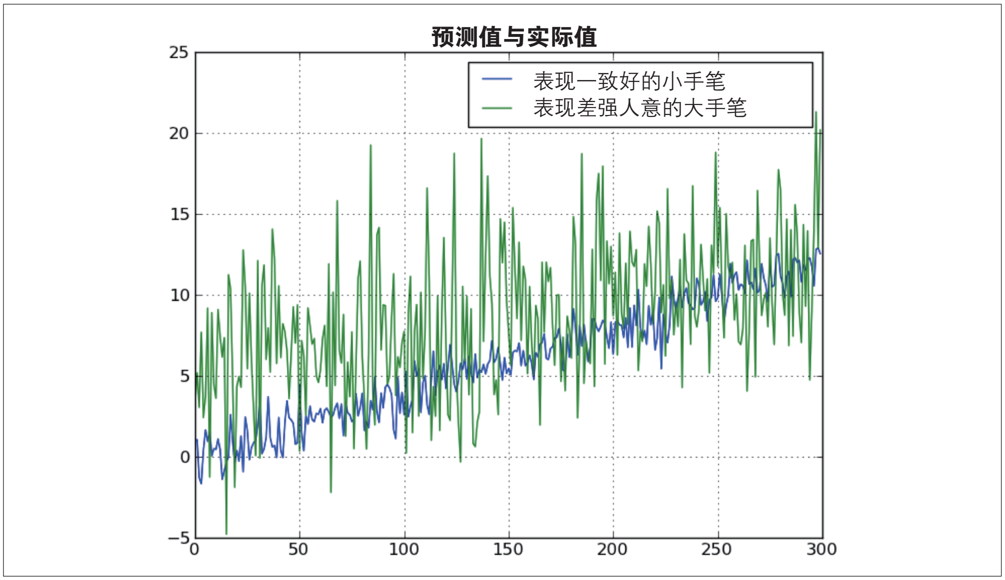


图 6-13：两个理论模型的累积 PnL 值对比图

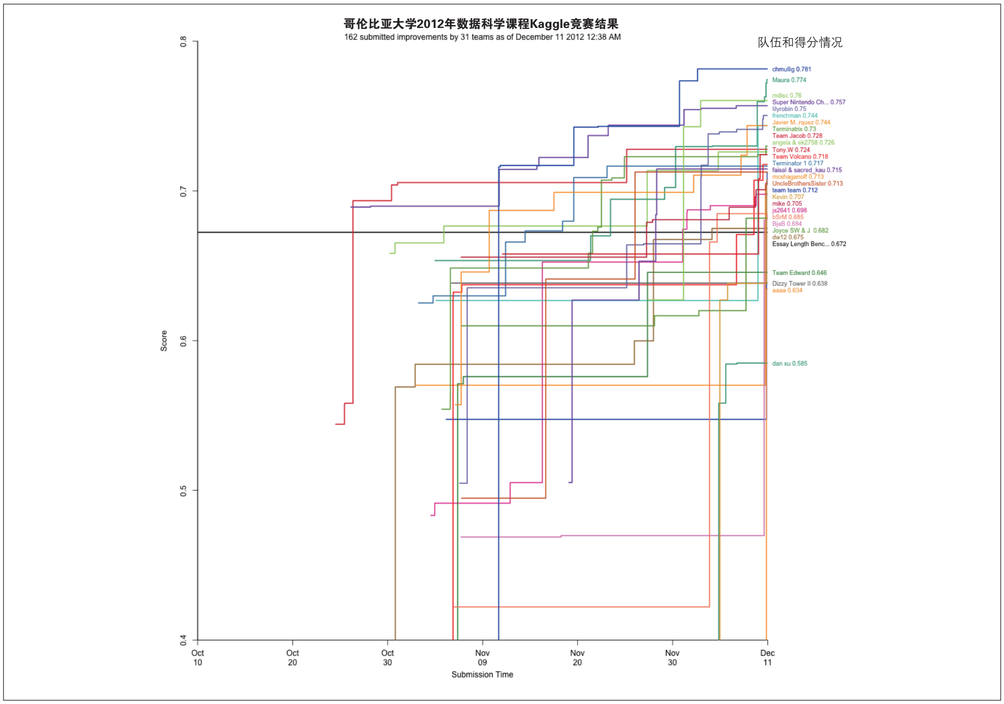


图 7-1：该图出自 Chris Mulligan，他是 Rachel 班里的学生。该图很好地描述了每个参赛个人 / 队伍在比赛期间，模型的进化情况

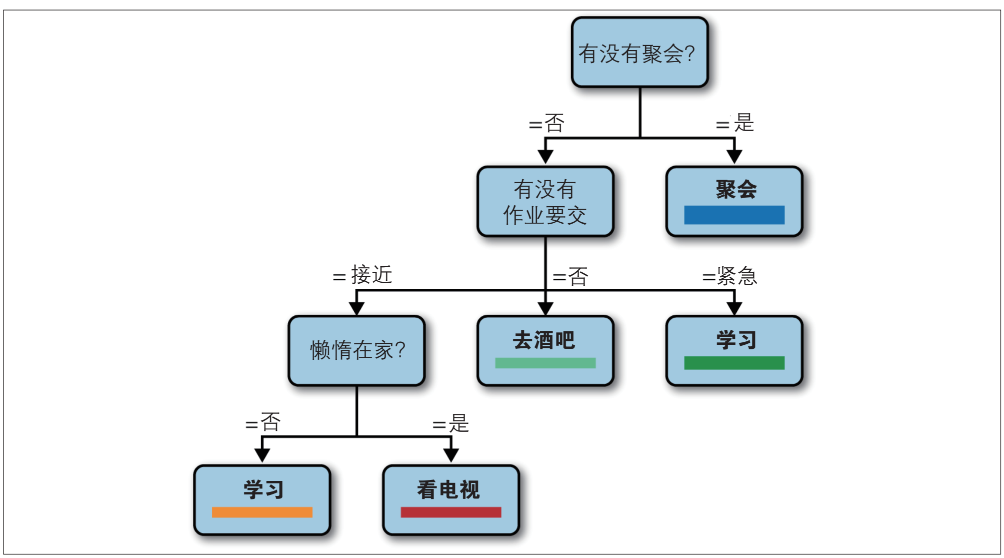


图 7-3：一个大学生在解决自己的时间分配问题时用到的决策树（原图摘自 Stephen Marsland 的著作 *Machine Learning: An Algorithmic Perspective*（《基于算法的机器学习》，Chapman and Hall/CRC），并获得了作者的许可

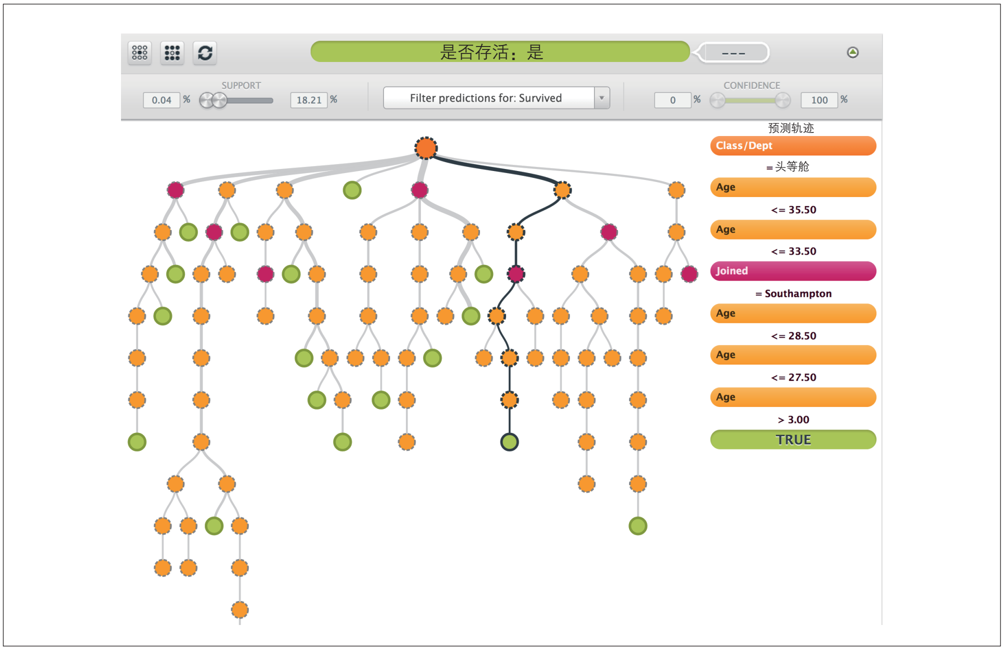


图 7-6：泰坦尼克号乘客生存模型的决策树图

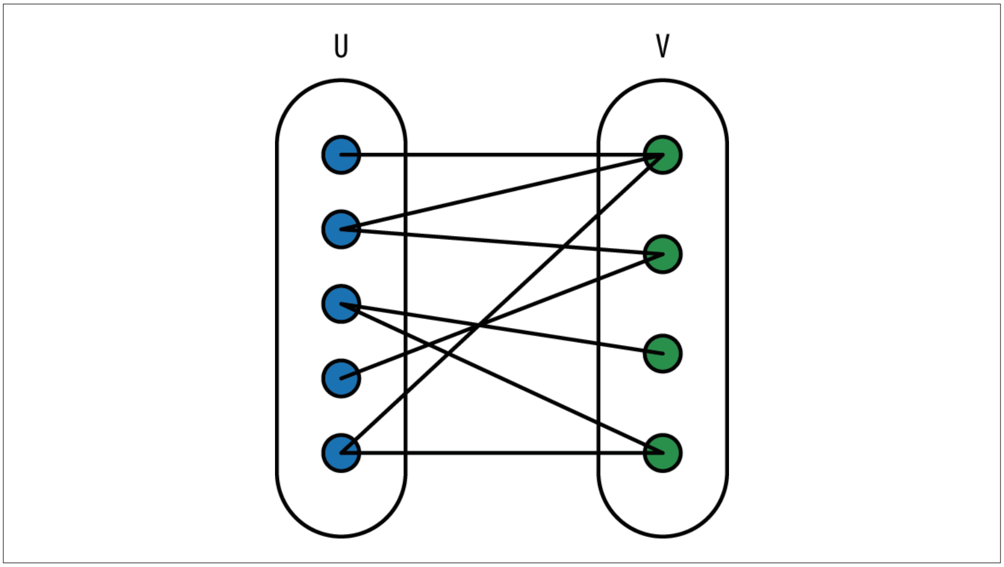


图 8-1：推荐系统的二分图：左侧是用户，右侧是推荐的项目，比如电视节目


Nuage Vert, HeHe (Helen Evans & Heiko Hansen)

图 9-1：Helen Evans 和 Heiko Hanse 的 Nuage Vert 可视化案例（http://youtu.be/l_4rTQCWItw）



Dusty Relief, New Territories

图 9-3：New Territories 的 Dusty Relief 项目（http://www.new-territories.com/roche2002bis.htm）

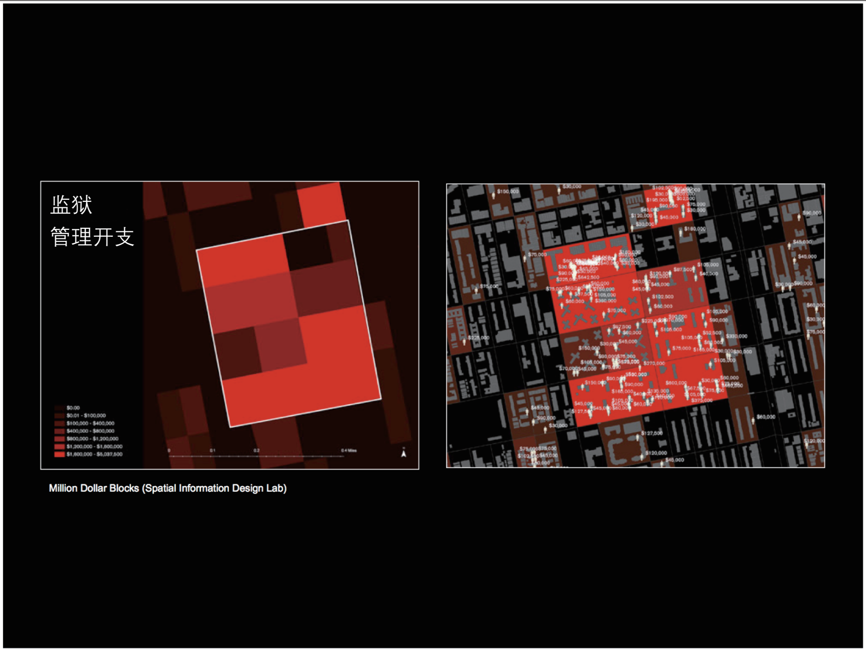


图 9-5：地理信息设计实验室（SIDL）设计的“百万美元街区”项目（http://www.spatialinformationdesignlab.org/）

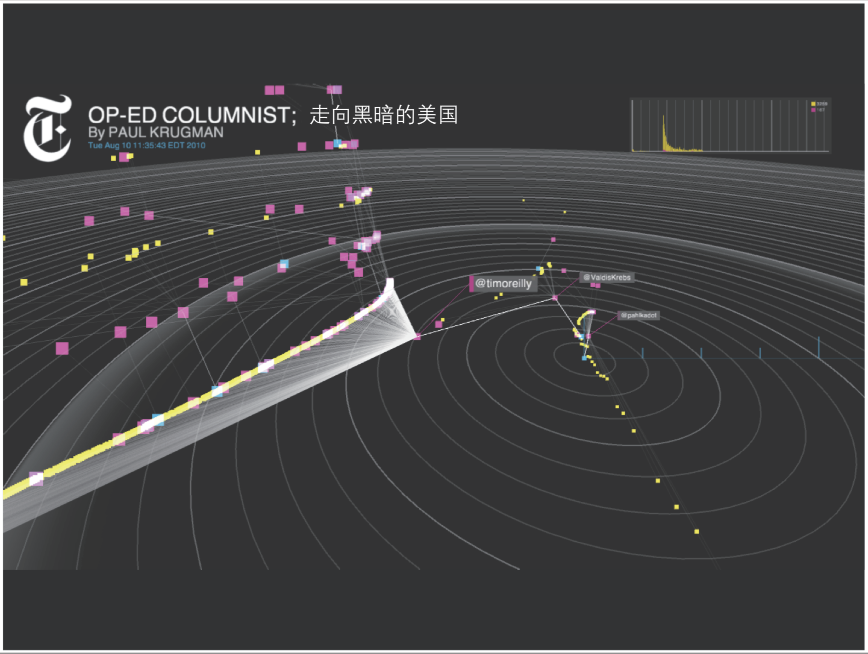


图 9-8：Jer Thorp 和 Mark Hanse 的作品：Cascade 项目

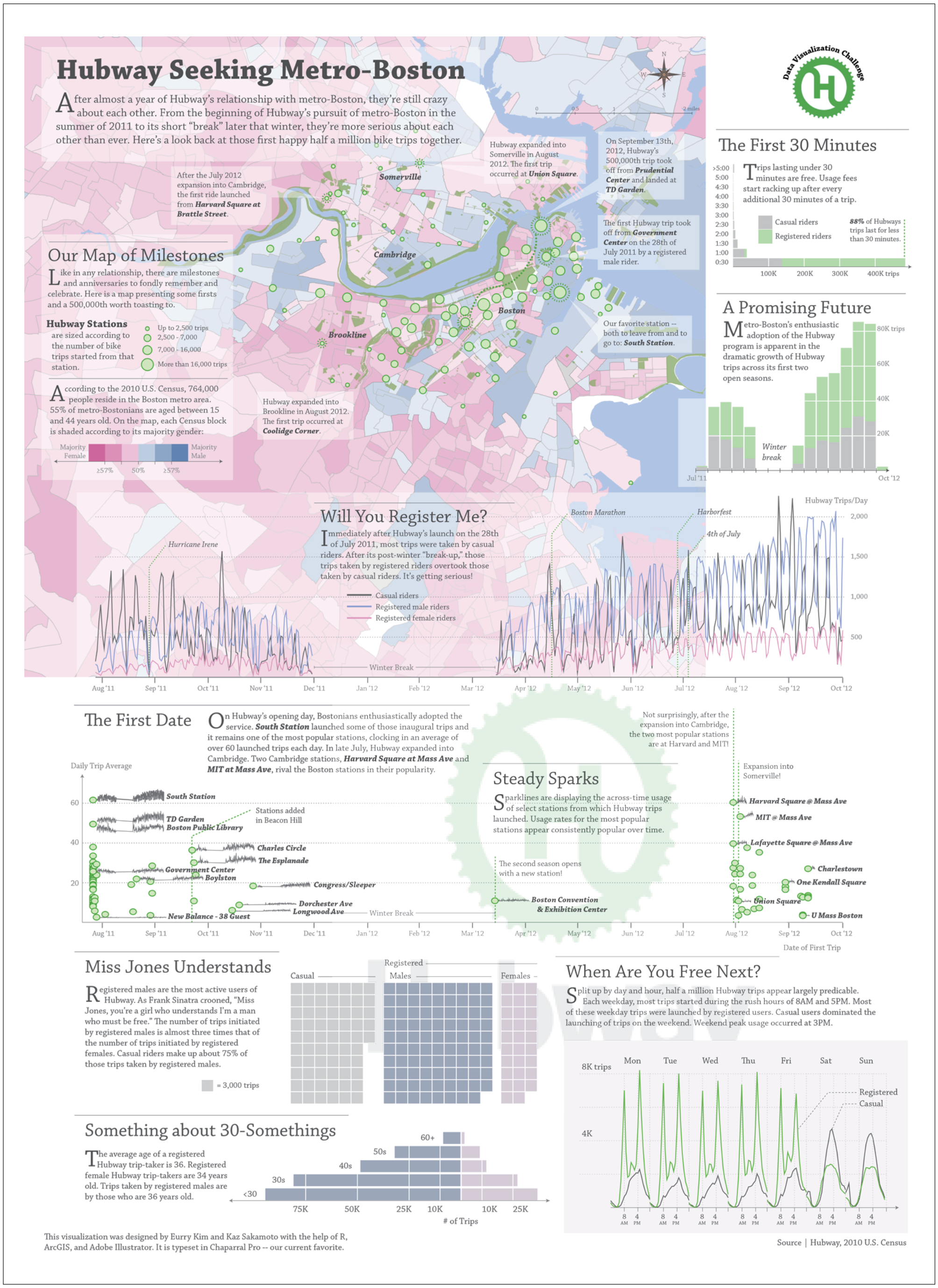


图 9-17：Eurry Kim 和 Kaz Sakamoto 参加 Hubway 公共自行车项目可视化竞赛的最终作品，以及这个项目在波士顿中心区实施的情况

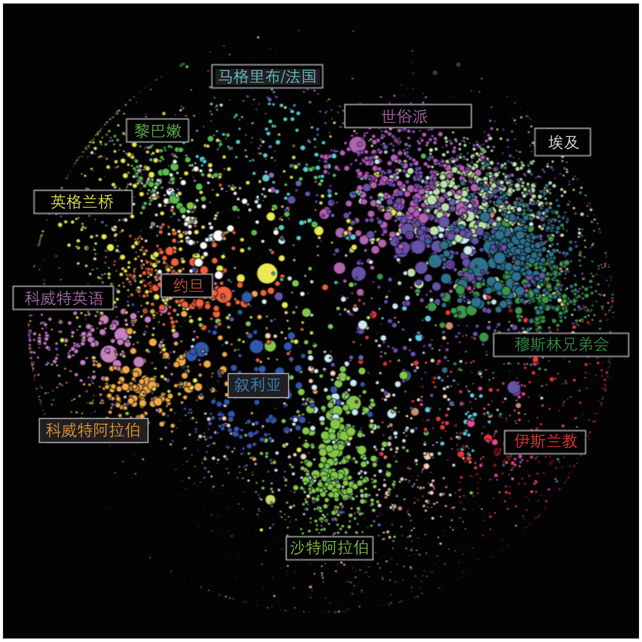


图 10-1：阿拉伯博客圈

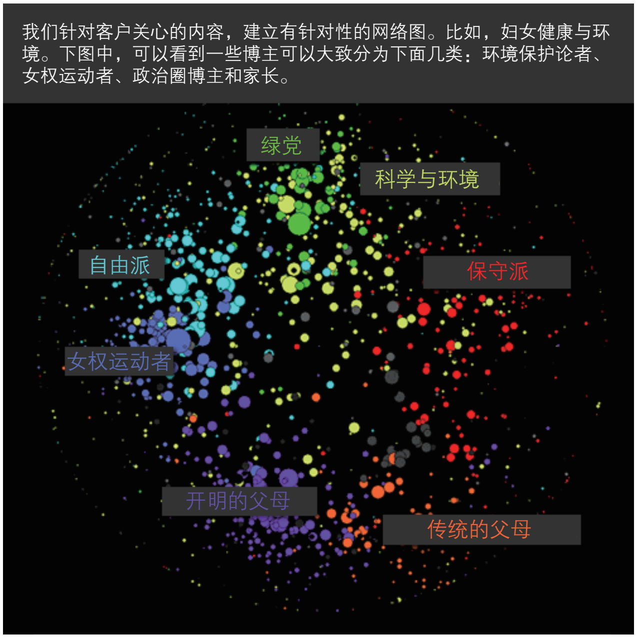


图 10-2：英语圈博客分类图

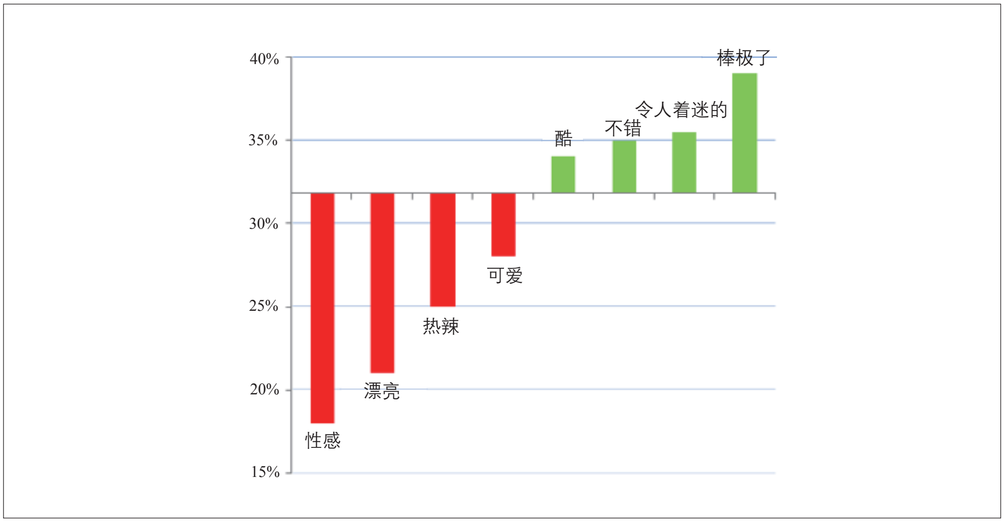


图 11-2：OK Cupid 的研究发现，在第一次接触性对话中使用“漂亮”一词不利于得到积极的答复

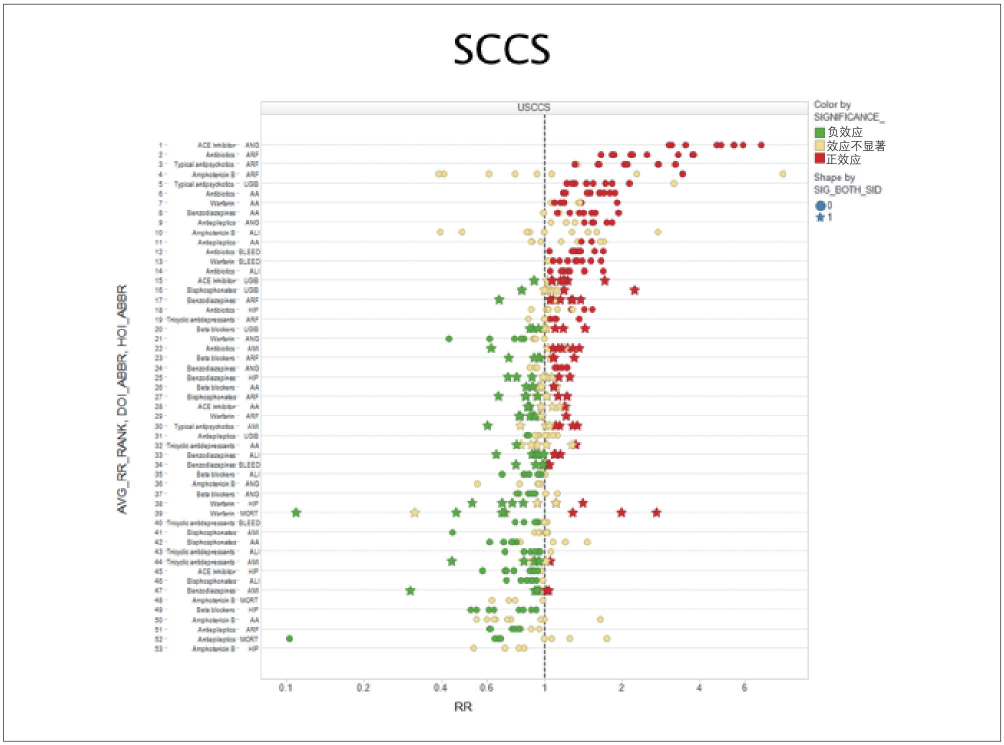


图 12-1：50 个课题的统计显著性分布图

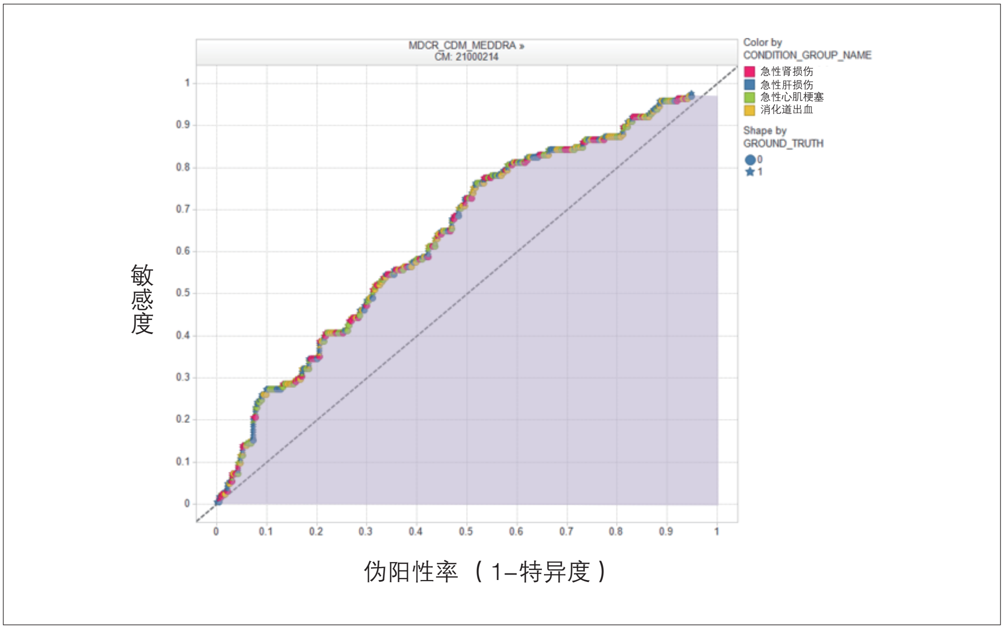


图 12-2：价值 2500 万美元的 ROC 曲线图

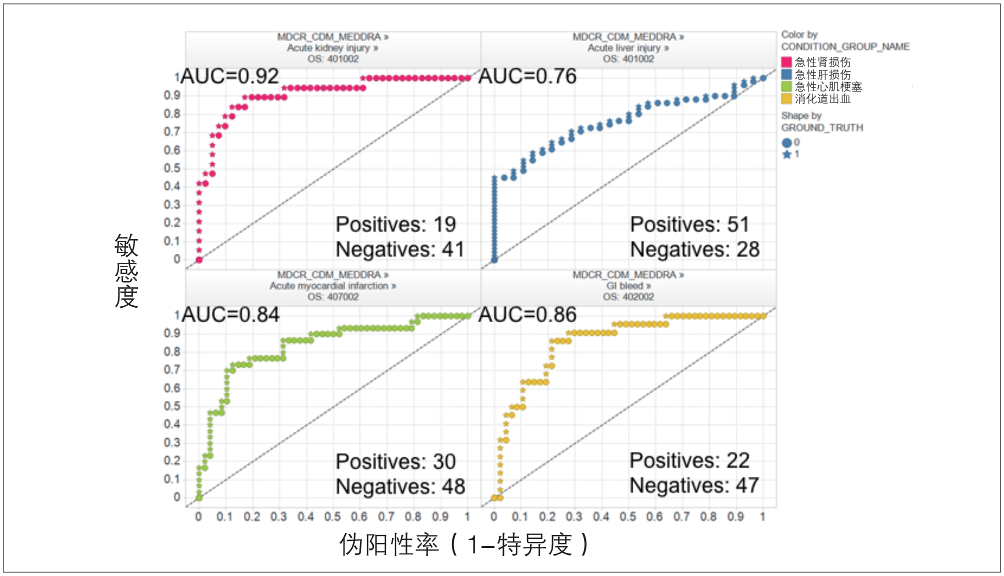


图 12-3：针对性的选取模型后，模型的预测效果可以得到较大改善

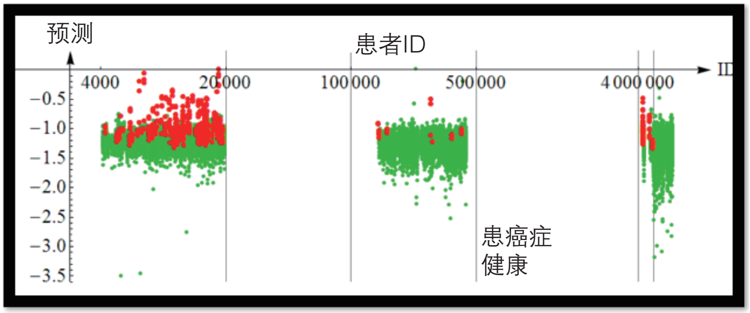


图 13-1：依患者的 ID 排序，红色代表得癌症的患者，绿色代表未得癌症的人

icd1x	icd2x	icd3x	icd4x
786	285	459	-1
401	486	-1	-1
401	486 ← 780	-1	-1
599	-1	-1	-1
V22	650	-1	-1
V56	492	586	-1
786	493	285	459

→

icd1x	icd2x	icd3x	icd4x
786	285	459	-1
401	-1	-1	-1
401	780	-1	-1
599	-1	-1	-1
V22	650	-1	-1
V56	492	586	-1
786	493	285	459

图 13-2：INFORMS 竞赛中数据是如何准备的

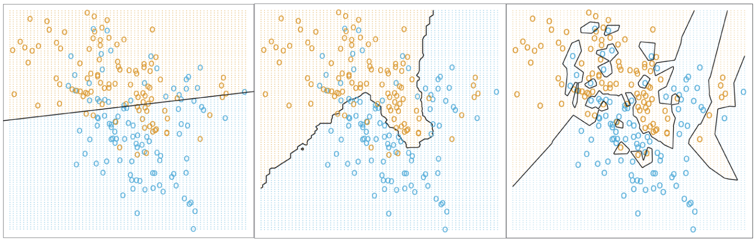


图 13-3：这幅经典的图片来摘自 Hastie 和 Tibshirani 合著的 *Elements of Statistical Learning*（《统计学习基础》，Springer-Verlag，参见 http://stanford.io/17szrYz），展示了同一份数据，对二值响应拟合线性回归模型时，采用 15 个最近邻和 1 个最近邻得到的不同结果

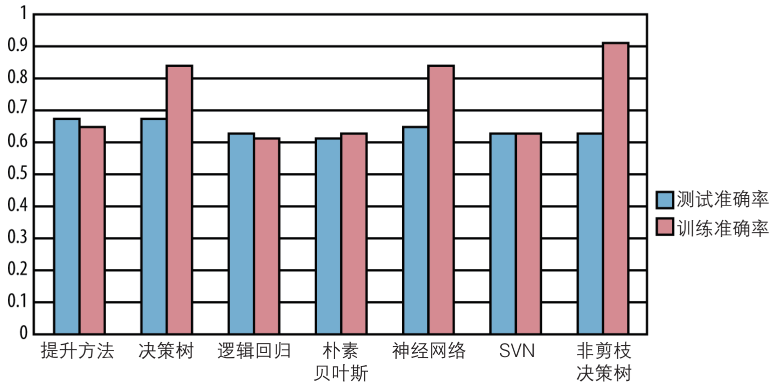


图 13-4：模型的差别

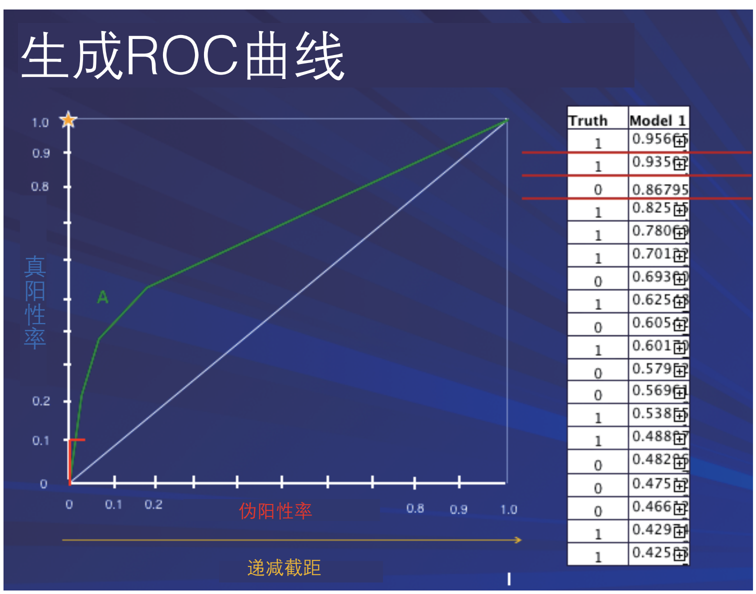


图 13-5：一个绘制 ROC 曲线的例子

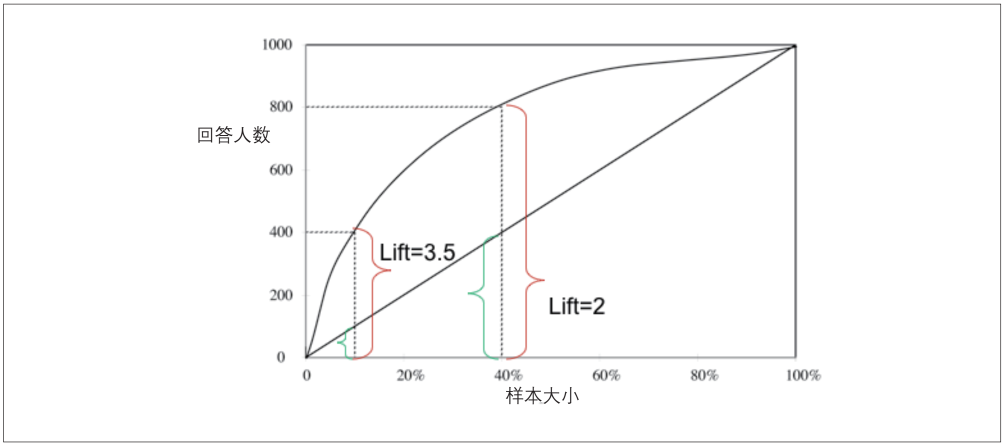


图 13-6：升力曲线

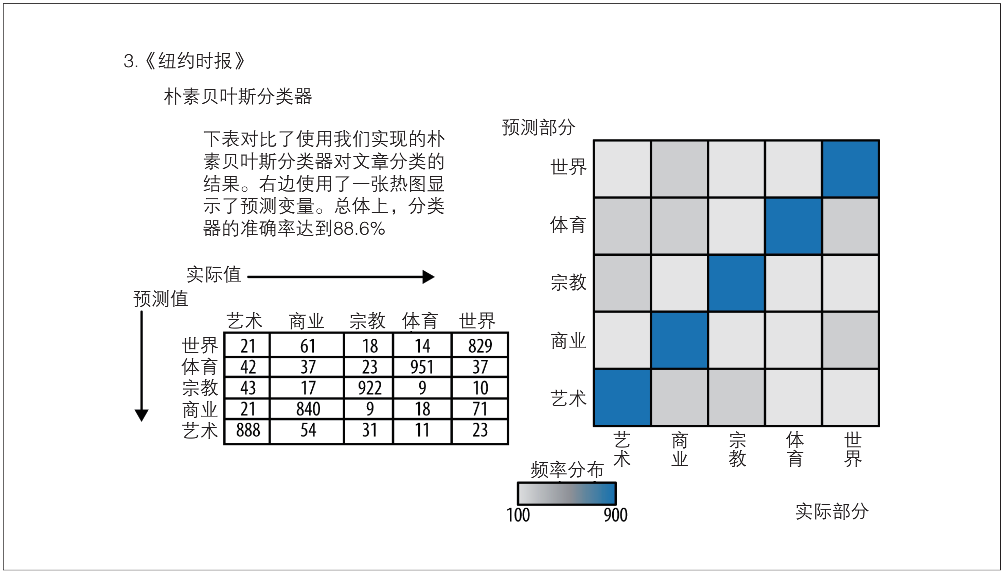


图 15-1：一位同学的部分答案

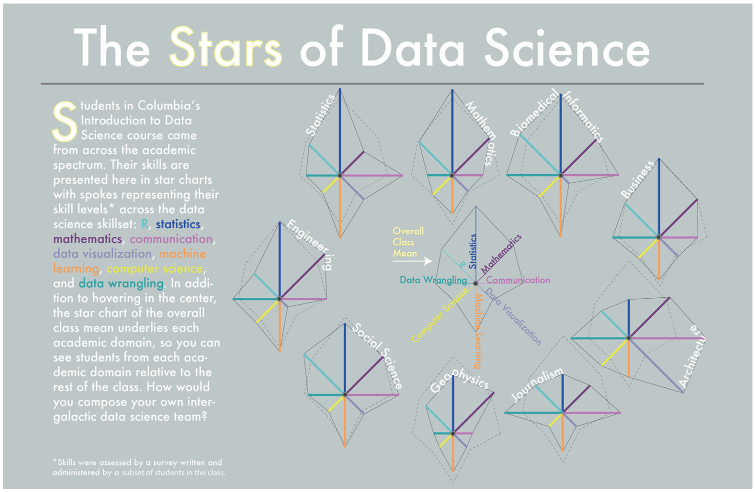


图 15-3：数据科学技能星型图（由 Adam Obeng、Eurry Kim、Christina Gutierrez、Kaz Sakamoto、Vaibhav Bhandari 合作完成）

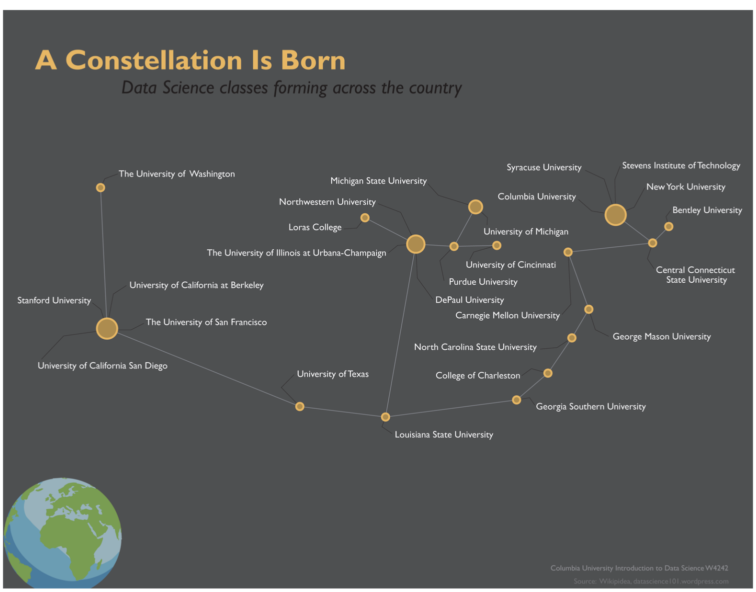


图 15-4：数据科学在各大学间流行程度的星座图（由 Kaz Sakamoto、Eurry Kim、Vaibhav Bhandari 合作完成）

TURING 图灵程序设计丛书

数据科学实战

DOING DATA SCIENCE

[美] Rachel Schutt Cathy O'Neil 著
冯凌秉　王群锋 译

O'REILLY®
Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo
O'Reilly Media, Inc.授权人民邮电出版社出版

人民邮电出版社
北　京

图书在版编目（CIP）数据

数据科学实战 / （美）舒特（Schutt,R.），（美）奥尼尔（O'Neil,C.）著；冯凌秉，王群锋译. -- 北京：人民邮电出版社，2015.3
（图灵程序设计丛书）
ISBN 978-7-115-38349-5

Ⅰ. ①数… Ⅱ. ①舒… ②奥… ③冯… ④王… Ⅲ. ①数据管理 Ⅳ. ①TP274

中国版本图书馆CIP数据核字(2015)第013752号

内 容 提 要

本书脱胎于哥伦比亚大学“数据科学导论”课程的教学讲义，它界定了数据科学的研究范畴，是一本注重人文精神，多角度、全方位、深入介绍数据科学的实用指南，堪称大数据时代的实战宝典。本书旨在让读者能够举一反三地解决重要问题，内容包括：数据科学及工作流程、统计模型与机器学习算法、信息提取与统计变量创建、数据可视化与社交网络、预测模型与因果分析、数据预处理与工程方法。另外，本书还将带领读者展望数据科学未来的发展。

本书适合所有希望通过数据分析解决问题的人阅读参考，包括数据科学家、金融工程师、统计学家、物理学家、学生及其他对数据科学感兴趣的人。

◆ 著　[美] Rachel Schutt Cathy O'Neil
译　冯凌秉　王群锋
责任编辑　李松峰　毛倩倩
执行编辑　周宇宁
责任印制　杨林杰

◆ 人民邮电出版社出版发行　北京市丰台区成寿寺路11号
邮编　100164　电子邮件　315@ptpress.com.cn
网址：http://www.ptpress.com.cn
北京　　印刷

◆ 开本：800×1000　1/16
印张：19.75　彩插：8
字数：487千字　2015年3月第1版
印数：1-4 000册　2015年3月北京第1次印刷
著作权合同登记号　图字：01-2014-5612号

定价：79.00元

读者服务热线：(010)51095186转600　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京崇工商广字第 0021 号

版权声明

O’Reilly Media, Inc.介绍

O’Reilly Media 通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自 1978 年开始，O’Reilly 一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O’Reilly 的发展充满了对创新的倡导、创造和发扬光大。

O’Reilly 为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了 Make 杂志，从而成为 DIY 革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O’Reilly 的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O’Reilly 现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项 O’Reilly 的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O’Reilly Radar 博客有口皆碑。”

——*Wired*

“O’Reilly 凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——*Business 2.0*

“O’Reilly Conference 是聚集关键思想领袖的绝对典范。”

——*CRN*

“一本 O’Reilly 的书就代表一个有用、有前途、需要学习的主题。”

——*Irish Times*

“Tim 是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照 Yogi Berra 的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去 Tim 似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——*Linux Journal*

目录

作者介绍

Rachel Schutt 是美国新闻集团旗下数据科学部门的高级副总裁。她从哥伦比亚大学取得博士学位后，加入谷歌研究院工作了数年。她是哥伦比亚大学统计系的兼职教授，同时也是哥伦比亚大学数据科学及工程研究所的教育委员会发起者之一。她有几个专利正在申请之中，这些专利基于她在谷歌的工作，在那里她设计了算法原型，并且通过建模来理解用户的行为，这些最终都反映在了直接面向用户的产品中。她同时拥有纽约大学的数学硕士学位、斯坦福大学的工程经济系统和运筹学硕士学位，以及密歇根大学的数学学士学位。

Cathy O'Neil 从哈佛大学获得了数学博士学位，她曾经是麻省理工数学系的博士后、巴纳德学院的教授（在那里她发表了大量算术代数几何方面的论文）。随后她转身投入工业界，先是在信贷危机中期加入了 D.E. Shaw 公司，担任该公司对冲基金的金融师；然后又加入了 RiskMetrics（一家对银行和对冲基金进行风险评估的软件公司）。她现在是纽约初创公司联盟的数据科学家，并且撰写和维护着一个博客 mathbabe.org。另外，她还参与了占领华尔街运动。

关于封面图

本书封面上的动物是九带犰狳（Dasypus novemcinctus），是一种广泛分布于中北美及南美洲的哺乳动物。在拉丁文中，novemcinctus 的字面意思是“九条带子”（位于腹部可伸缩甲壳后方），实际上九带犰狳身上的“带子”数量通常为 7~11 条。产自南美洲的三带犰狳是唯一一种在遇到危险时可将身体团成球状来保护自身的犰狳，其他种类的犰狳则由于甲壳太多无法做到这点。

犰狳的皮肤最为惹人注目，它的皮肤呈灰褐色、皮革状，由叫作鳞甲的鳞片组成，覆盖了除身下外的其他部分。犰狳长有利爪，善于挖洞，通常在自己的领地内挖好几个洞，并且用臭腺给自己的洞穴做记号。九带犰狳的体重通常为 5.5~14 磅，大小和一只大点的家猫差不多。犰狳以昆虫为主食，但是也吃水果、小爬虫和动物的卵。

雌性犰狳一般一胎生四个幼崽，四个幼崽性别都是一样的，这是因为雌性犰狳受精后，受精卵分裂成了四个胚胎。犰狳刚出生时皮肤异常柔软，长大后皮肤慢慢变得坚硬起来。犰狳在出生几小时后即可爬行。

九带犰狳受到惊吓后，可跳起 3~4 英尺。虽然这一应激反应能吓走自然界中的食肉动物，但是如果面对一辆行驶而来的汽车，这个反应却是致命的：犰狳会撞上迎面而来的汽车。犰狳和人类还有另外一种不幸的联系，它是唯一已知的麻风病病毒携带者，人类在食用或接触过犰狳后感染上麻风病的消息并不鲜见。

封面图取自英国动物学家乔治 · 肖的动物学图谱，由 Karen Montgomery 重新着色。

前言

Rachel Schutt

2012 年秋天，我在哥伦比亚大学开设了一门新课：数据科学导论。作为一个新兴领域，数据科学在学术界尚未划分为一个独立学科。那么数据科学到底是什么呢？我将这门课的讲义集结成书，试图回答这一问题。

为了帮助读者理解本书及其缘起，我觉得有必要简单介绍一下我自己，和我设计并讲授这门课的初衷。

初衷

简单地说，我期望在我上大学时就有这样的课。但那是 20 世纪 90 年代，数据爆炸尚未开始，开设这样一门课也就无从谈起。我本科时主修数学专业，主要是做理论和实证研究。虽然很庆幸这些训练赋予了我严谨解决问题的能力，但同时我也略感遗憾，若当时能再学点实际应用的技巧就更好了。

在从大学毕业到获得统计学博士学位期间，我走了一些弯路，我一直在试图寻找适合自己的研究领域，喜欢探究隐藏在宇宙中的模式，喜欢解答有趣的谜题，希望可以将自己的这些爱好物尽其用。之所以谈起这些，是因为现在很多学生觉得必须先知道自己这辈子到底想要干什么，我做学生时，不可能规划将来要从事数据科学相关的工作，因为那时根本还没有数据科学这样一个领域。因此我建议这些学生，或者其他愿意听我在这儿唠叨的人：大可不必这样。不必现在就规划好未来，走点弯路也没什么，谁知道这一路上你会发现什么呢？我拿到统计学博士学位后，在谷歌工作了几年，在这几年中，数据科学、数据科学家这些术语才在硅谷流行起来。

这个世界有许多问题尚未解决，对于那些拥有量化思维又乐于开动大脑的人来说，在解决问题的过程中充满了机遇。我的目标是帮助学生们成为具有批判性思维的人、能用创新思

维去解决问题（甚至是人们尚未发现的问题）的人、对世界充满好奇喜欢问问题的人。若要我去构建一个数学模型，去为治愈癌症贡献一份力量，或者揭示出自闭症的奥秘，或者用来预防恐怖袭击，我或许永远做不到。但我的学生有一天会做到，我教给了他们这些知识，就算完成了自己的使命。写作此书，使我有机会将毕生所学传播给更多的人，我希望他们能从中得到激励，或者学到一些有用的工具，来让这个世界变得更好，而不是更坏。

建模和数据分析的过程并非彻底地中立，会受到研究者个人价值观的影响。研究的问题是由你来挑选的，研究假设也是你根据模型得出的，度量方法和算法也是由你来设计的。

世界上也并不是所有的问题都需要用数据科学或技术手段来解决，一个好的数据科学家是指他能甄别出哪些问题适合用数据科学解决，构建出对应的数据模型或者编写代码去解决它。但是我相信，在多学科的团队中，如果有一个理解数据、具有量化思维、精通编程的问题解决者（让我们将这种人称为“数据科学家”），这个团队可能会走得更远。

课程的起源

我在 2012 年 3 月份提议开设此课，主要原因有三。其中第一个原因最重要，我将会花最大篇幅去阐述。

原因一：我想告诉我的学生业界的数据科学家是怎么工作的，并且让他们掌握一些数据科学家所使用的技术。

在为 Google+ 工作时，我所在的数据科学团队由一群身怀绝技的博士组成，其中有学社会学的、学工程的、学物理的和学计算机的，而我是统计学专业的。我们隶属于一个更大的团队，这个团队有很多天才的数据工程师，他们实现数据管道、基础架构、分析面板和一些实验性质的架构（用来做 A/B 测试）。我们的团队架构是扁平化的，我们有海量的数据，每个人都是各自领域的专家，我们精诚合作，做出了很多不可思议的事，包括建立预测模型、实现算法原型、揭示出隐藏在数据背后的模式，这些对我们的产品影响深远。

以数据为基础，我们为领导层的决策提供真知灼见；分析因果关系，我们发展出了新的方法论。这些全仰仗世界一流的工程师和技术设备。每个人都为团队引入了专家级的技能，包括编码、软件工程、统计学、数学、机器学习、通信、可视化、探索性数据分析（EDA）等，还有对社交网络和社交空间的数据的敏感直觉和专业知识。

要知道，没有人是全知全能的，但集合所有人的智慧，我们就做到“无所不能”。我们认识到了每种技能的价值，因此就成功了。我们的共同点是守信，对解决有趣的问题充满好奇心，对待新的科学发现既保有适度的怀疑又充满激情。我们喜爱这项工作，对数据背后的模式充满了好奇。

我居住在纽约，希望把我在谷歌公司的工作经验传授给哥伦比亚大学的学生们，我相信他

们需要这个，而且，我也喜欢教学。我想把我从工作中学到的东西教给他们。另外，我知道纽约的技术圈里有一个新兴的数据科学家社区，我也希望学生们能从他们身上汲取知识。

因此，这门课程常会邀请业界或学术界的数据科学家来做客座演讲。每位嘉宾所专长的技能和领域都不尽相同。我希望通过这样一种多样性的组合，让学生们对数据科学有一个更全面的认识。

原因二：数据科学有希望成为一门极具研究价值、意义深远的学科，它会影响到人们生活的方方面面。为此，哥伦比亚大学和纽约市市长布隆伯格先生在 2012 年 7 月宣布成立了一个数据科学与工程研究所。开设这门课是在尝试发展数据科学的理论，我希望让数据科学成为一门真正的科学。

原因三：我时常听到业界的数据科学家说，在脱离实践的课堂上是无法真正教授数据科学的，我想挑战一下这种言论。我一直将我的课堂视作数据科学家的孵化器，而我的学生也确实表现出色，他们将会成为数据科学界冉冉升起的新星。事实上，本书其中一章内容就是由我的学生们贡献的。

本书的起源

如果不是遇到了 Cathy O'Neil，我的教学笔记也不会集结成书。她是一位数学家，后来转型为数据科学家，她的个人博客 mathbabe.org 很受欢迎，在博客中的"关于自己"部分，她说自己一直在期待下面这个问题能有更好的答案：非理论派的数学家能做些什么以让这个世界变得更加美好？我向大学提议开设数据科学导论这门课程时，恰好认识了 Cathy，那时她正在一个初创公司工作，职位是数据科学家。对于我开课的尝试，她十分支持。她还提出亲自过来听课，并在博客上同步直播我的授课内容。鉴于我性格比较内向低调，起先我并不喜欢这么做，后来 Cathy 说服了我。她说这与商业广告的肆意炒作截然不同，这是一个绝好的机会，借此可以将"数据科学"的概念向大众普及。

我在哥伦比亚大学上的每一节课，Cathy 都会坐在第一排，并不时提出问题。她后来还受邀作为这门课的客座嘉宾给同学们上了一课（见第 6 章）。除了将我的讲义发布到博客上，Cathy 还对授课内容贡献甚巨，比如，她提醒我们数据建模过程中存在一些道德伦理方面的考量。此外，她鼓励我也同步开设一个博客（http://columbiadatascience.com/blog/），用来和学生们做直接交流。我在上面也会总结自己的教学经验，这或许会帮到其他教授。Cathy 博客中所有关于我授课内容的条目，再加上我博客中的部分内容，构成了本书的原始素材，我们在这一基础上修改加工，再集合一些其他资料，终成此书。

本书内容

本书既介绍实践应用，也提出理论规范。一方面，本书介绍了一些业内顶尖数据科学家的

日常工作内容，带大家看看他们在实践中如何应用数据科学知识，借此管中窥豹，了解这一学科目前的应用现状。另一方面，我们还将从学术角度去定义数据科学的研究范畴。

这不是一本关于机器学习的教科书。恰恰相反，本书会多角度、全方位、深入地介绍数据科学。它是对现有数据学科领域的纵览，试图为这一学科勾勒出一幅全景图。因此，在选择案例时，我们会更注重广度而非深度。

希望本书能够被那些善待它的人充分利用，举一反三，去解决那些重要的问题。

这门课在哥伦比亚大学讲完后，我听到了这样的评价：它是一门从人文主义角度全面讲解数据科学的课程。我们不仅关注工具、数学、模型、算法和代码，同时也很关注上述过程中的人性化考量。关于什么是人文主义者，我很喜欢如下的定义："他十分关心人类的福祉，尊重个人的价值观，并且注重维护个体尊严。"如何在数据科学中体现人文主义？你在建模和设计算法时，认识到你作为个人所应起到的作用，想想哪些东西是人所具备而电脑不具备的，比如基于道德的判断；向世界公布一种新的统计模型前，想想会为他人的生活带来什么样的影响。

组织结构

本书的组织结构遵循我在哥伦比亚大学的数据科学导论课程，在第 1 章，我们将会回答"什么是数据科学"这个核心问题，同时介绍数据科学工作流程，这是全书组织结构的纲领。第 2 章和第 3 章对统计模型和机器学习算法做一概览，它们是后续章节的基础。第 4 章到第 6 章，以及第 8 章将会针对特定案例深入学习一些模型和算法。第 7 章讲述如何从数据中提取有效信息以及在模型中创建统计变量。第 9 章和第 10 章将深入介绍一些传统学术界很少涉足的内容（当然现在情况有所改善）：数据可视化和社交网络。第 11 章和第 12 章将从预测模型转而介绍因果分析。第 13 章和第 14 章介绍数据预处理以及工程方法。第 15 章是我的学生们讲述他们的故事——他们是怎样学习数据科学的。第 16 章展望数据科学未来的发展。

阅读须知

阅读本书时最好从前往后依序阅读，这样更便于理解，因为不少概念都是一环扣一环的。如果你的统计和概率背景不强，或者从前没有编过程，那么阅读本书的同时，如能阅读本章末尾附带的补充材料以查漏补缺，效果将会更好。全书为大家推荐了很多补充材料，当你阅读某个章节感到困难时，这或许由于你缺失某些背景知识，或许由于我们的讲解不够清晰，这时你都可以求助于这些补充材料，厘清概念。

书中的代码

本书不是一本手册，书中代码仅供示范，有时需要读者自己去亲手实践实现某些算法，以期对其中的概念理解得更加深入。

目标读者

虽然媒体把数据科学和数据科学家渲染成摇滚巨星一般，但别怕，数据分析学科的门槛并没有那么高。你是否是金融工程师并不重要，只要你热爱解决问题，热爱寻找数据背后的模式，那么数据科学就绝对是你的菜。

具备各种专业背景的读者均可阅读本书，我们希望每位读者都能各取所需，读出自己心中的“哈姆雷特”。

- 有经验的数据科学家能从一个全新的角度认识自己和自己从事的行业。
- 统计学家能从中了解到统计学和数据科学的关系。也许他们仍然坚持过去的老态度：什么数据科学，这就是统计学嘛。我们也不介意这样的争论。
- 金融、数学、物理等学科的博士们，若有意转行研究数据科学，或者只是希望学习一些数据科学技能，都能从本书中了解到他们需要学些什么。
- 学生或从未接触过数据科学的人，会觉得被直接扔进了该领域的深水区。如果不能一下子理解书中内容，请不要害怕，这是必经之路。
- 对于那些从未用 R 或 Python 写过程序的读者，我们推荐一本书：*The Art of R Programming*，Norman Matloff 著（No Starch Press），在哥伦比亚大学选修过该课程的学生们也从实验讲师 Jared Lander 的讲解中获益良多，他的新书 *R for Everyone: Advanced Analytics and Graphics*（Addison-Wesley）已于 2013 年 12 月上市。书中所有练习也可以使用 Python 的各种工具包实现。
- 对于那些完全没有编程经验的读者，上述建议也适用，但可能需要先阅读一些基础的编程书籍，如下两本书就是不错的选择：Mark Lutz 和 David Ascher 的 *Learning Python*（O’Reilly），以及 Wes McKinney 的 *Python for Data Analysis*（O’Reilly）。

基础知识要求

我们假设本书读者学过线性代数、概率论和统计学，还有一些编程经验。不过，没有学过也没关系，我们试图让本书在内容上能够尽可能涵括所有内容。当然，如果读者在某方面知识欠缺，大可以针对不足自行寻找其他补充阅读材料。我们在全书中也都尽可能向读者提示哪些补充阅读材料能让你更深刻地理解相关问题。

补充阅读

数据科学作为一个新兴领域，植根于其他学科：统计推断、算法、统计模型、机器学习、实验设计、优化理论、概率论、人工智能、数据可视化和探索性数据分析等。每门学科都值得花好几门课或好几本书专门讲解，这正是写作本书面临的一个极大挑战，因此我们将这些补充阅读材料附在后面，希望读者在需要时可以参考。

数学

- *Linear Algebra and Its Applications*，Gilbert Strang 著（Cengage Learning）
- *Convex Optimization*，Stephen Boyd 和 Lieven Vendenberghe 著（Cambridge University Press）
- *A First Course in Probability*（Pearson）、*Introduction to Probability Models*（Academic Press），Sheldon Ross 著

编程

- *R in a Nutshell*，Joseph Adler 著（O’Reilly）
- *Learning Python*，Mark Lutz 和 David Ascher 著（O’Reilly）
- *R for Everyone: Advanced Analytics and Graphics*，Jared Lander 著（Addison-Wesley）
- *The Art of R Programming: A Tour of Statistical Software Design*，Norman Matloff 著（No Starch Press）
- *Python for Data Analysis*，Wes McKinney 著（O’Reilly）

数据分析与统计推断

- *Statistical Inference*，George Casella 和 Roger L. Berger 著（Cengage Learning）
- *Bayesian Data Analysis*，Andrew Gelman 等著（Chapman & Hall）
- *Data Analysis Using Regression and Multilevel/Hierarchical Models*，Andrew Gelman 和 Jennifer Hill 著（Cambridge University Press）
- *Advanced Data Analysis from an Elementary Point of View*（http://goo.gl/udICRX），Cosma Shalizi 著（Cambridge University Press）
- *The Elements of Statistical Learning: Data Mining, Inference and Prediction*，Trevor Hastie、Robert Tibshirani 和 Jerome Friedman 著（Springer）

人工智能和机器学习

- *Pattern Recognition and Machine Learning*，Christopher Bishop 著（Springer）
- *Bayesian Reasoning and Machine Learning*，David Barber 著（Cambridge University Press）
- *Programming Collective Intelligence*，Toby Segaran 著（O’Reilly）
- *Artificial Intelligence: A Modern Approach*，Stuart Russell 和 Peter Norvig 著（Prentice Hall）
- *Foundations of Machine Learning*，Mehryar Mohri、Afshin Rostamizadeh 和 Ameet Talwalkar 著（MIT Press）
- *Introduction to Machine Learning (Adaptive Computation and Machine Learning)*，Ethem Alpaydim 著（MIT Press）

实验设计

- *Field Experiments*，Alan S. Gerber 和 Donald P. Green 著（Norton）
- *Statistics for Experimenters: Design, Innovation, and Discovery*，George E. P. Box 等著（Wiley-Interscience）

可视化

- *The Elements of Graphing Data*，William Cleveland 著（Hobart Press）
- *Visualize This: The FlowingData Guide to Design,Visualization*，*and Statistics*，Nathan Yau 著（Wiley）

关于本书其他贡献者

我邀请了很多嘉宾来我的数据科学导论课上做讲座，他们有的来自初创公司、科技企业，当然还有人就是我所在大学的教授。书中多章的内容都是在这些讲座的基础上写成的。虽然他们没有执笔写作此书，但书中很多内容及想法皆来自他们，此外，他们还帮助我审阅了各自讲座所对应章中的内容，并给出了很好的建议，对此我们深表谢意。没有他们我的数据导论课就不会成功，没有他们也不会有这本书，他们是我在数据科学领域的楷模。

排版约定

本书使用了下述排版约定。

- 楷体
  表示新的术语。

- 等宽字体（`constant width`）
  表示程序片段，也用于在正文中表示程序中使用的变量、函数名、命令行代码、环境变量、语句和关键字等代码文本。

- 加粗的等宽字体（**`constant width bold`**）
  表示应该由用户输入的命令或者其他文本。

- 斜体的等宽字体（*`constant width italic`*）
  表示应该由用户输入的值或根据上下文决定的值替换的文本。

这个图标代表小技巧、建议或说明。

这个图标代表警告或提醒注意的信息。

使用代码示例

你可以在这里下载本书随附的资料（数据集、练习题等）：https://github.com/oreillymedia/doing_data_science。

本书旨在帮助读者解决实际问题。一般来说，对于书中提到的代码，也许你需要在自己的程序或文档中用到，但除非大段大段地使用，否则你不必与我们联系取得授权。因此，用本书中的几段代码写成一个程序不用向我们申请许可。但是，销售或者传播 O'Reilly 图书随附代码的光盘必须事先获得授权。引用书中的代码来回答问题你也无需我们授权，将大段的示例代码整合到你自己的产品文档中则必须经过许可。

使用我们的代码时，希望你能标明它的出处。出处一般要包含书名、作者、出版社和 ISBN，例如：*Doing Data Science* by Rachel Schutt and Cathy O'Neil (O'Reilly). Copyright 2014 Rachel Schutt and Cathy O'Neil, 978-1-449-35865-5。

如果还有其他使用代码的情形需要与我们沟通，可以随时与我们联系：

permissions@oreilly.com

Safari® Books Online

Safari Books Online（http://www.safaribooksonline.com）是按需获取的数字图书馆。它同时以图书和视频的形式出版世界顶级技术和商务作家的专业作品。

技术专家、软件开发人员、Web 设计师、商务人士和创意专家等，在开展调研、解决问题、学习和认证培训时，都将 Safari Books Online 视作获取资料的首选渠道。

对于组织团体、政府机构和个人，Safari Books Online 提供各种产品组合和灵活的定价策略。用户可通过一个功能完备的数据库检索系统访问 O'Reilly Media、Prentice Hall Professional、Addison-Wesley Professional、Microsoft Press、Sams、Que、Peachpit Press、Focal Press、Cisco Press、John Wiley & Sons、Syngress、Morgan Kaufmann、IBM Redbooks、Packt、Adobe Press、FT Press、Apress、Manning、New Riders、McGraw-Hill、Jones & Bartlett、Course Technology 以及其他几十家出版社的上千种图书、培训视频和正式出版之前的书稿。要了解 Safari Books Online 的更多信息，我们网上见。

联系我们

请把对本书的意见和疑问发送给出版社。

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

北京市西城区西直门南大街 2 号成铭大厦 C 座 807 室（100035）
奥莱利技术咨询（北京）有限公司

O'Reilly 的每一本书都有专属网页，你可以在那儿找到本书的相关信息，包括勘误表、示例代码以及其他信息。本书的网站地址是：

http://oreil.ly/doing_data_science

对于本书的评论和技术性问题，请发送电子邮件到：

bookquestions@oreilly.com

要了解更多 O'Reilly 图书、培训课程、会议和新闻的信息，请访问以下网站：

http://www.oreilly.com

我们在 Facebook 的地址如下：http://facebook.com/oreilly

请关注我们的 Twitter 动态：http://twitter.com/oreillymedia

我们的 YouTube 视频地址如下：http://www.youtube.com/oreillymedia

致谢

感谢在谷歌的同事：David Huffaker、Makoto Uchida、Andrew Tomkins、Abhijit Bose、Daryl Pregibon、Diane Lambert、Josh Wills、David Crawshaw、David Gibson、Corinna Cortes、Zach Yeskel 和 Gueorgi Kossinetts。另外，还要感谢在哥伦比亚大学统计系的朋友们：Andrew Gelman 和 David Madigan，还有课程的实验指导者和助教 Jared Lander 和 Ben Reddy。

最后，也最真诚地感谢家人和朋友，他们提供了无尽的支持和爱，他们是：Eran Goldshtein、Barbara 和 Schutt、Becky、Susie 和 Alex、Nick、Lilah、Belle、Shahed，以及 Feeneys 一家。

——Rachel Schutt

感谢朋友与家人，尤其是可爱的儿子和丈夫，他们允许我每周为这门课程写一次博客。

——Cathy O’Neil

我们还要共同对以下人士和单位致谢。

- 在 Cathy 家参加聚会的那些志同道合的朋友：Chris Wiggins、David Madigan、Mark Hansen、Jake Hofman、Ori Stitelman 和 Brian Dalessandro。
- 编辑 Courtney Nash 和 Mike Loukides。
- IMA User 级建模会议的参与者及组织者，很多关于本书的基本构思即来源于那里。
- 学生们！
- Coppelia 餐馆，Cathy 和 Rachel 经常碰头和吃早餐的地方。

最后，我们将谢意诚敬给 Johnson 实验室的 John Johnson 和 David Park，感谢他们的慷慨大方，提供给我充足的时间用来写作此书。

第 1 章

简介：什么是数据科学

过去几年，“数据科学”和“大数据”的概念被媒体炒得热火朝天。对于这种现象，人们一开始难免疑惑，甚至怀疑。事实上，这就是 Cathy 和我当时的反应。

对于这些概念，Cathy 和我在很长一段时间里都感到迷茫，直到我们俩相识。我们一般会在星期三共进早餐，每当谈起这种现象，都有一种不安的感觉，总觉得在这喧嚣背后确然有一股新潮流在涌现，这股潮流或许是意义深远的，代表着我们整个文化范式在数据的影响下都会产生深刻的改变。Cathy 和我都是干这行的，觉得应该发挥我们的强项，去探索这些现象背后的原因，而不是置之不理。

在深入探索之前，我们有必要先介绍一下媒体所炒作的大数据时代，也许你和我们一样，也认为那些概念难以理解、语焉不详。然后，本章会进一步讲解我们是如何拨开迷雾发现背后的真相，以至于 Rachel 决定在哥伦比亚大学开设数据科学导论课程，而 Cathy 则在她的博客上同步记录该课程的内容，乃至上述所有内容终于结集成书送到你手中。

1.1　大数据和数据科学的喧嚣

让我们抛开炒作，因为很多人可能和我们一样，都对数据科学心存怀疑。之所以一上来就讲这些，是想让你知道：*我们也和你一样！*假如你也心存疑虑，说明你也很可能会贡献一份力量，推动数据科学的健康发展，使其对社会产生积极的影响，也使数据科学这门学科趋于正统，在众多学科中能占有一席之地。

让我们先来细数大数据和数据科学之所以这样让人如坠云里雾里的原因。

(1) 大多数基本的术语都缺乏严格定义。究竟什么是大数据？数据科学又是什么意思？大数据和数据科学之间有什么关系？数据科学就是关于大数据的科学吗？只有像谷歌和 Facebook 这样的高科技企业才用得到数据科学吗？为什么有人认为大数据是一个交叉学科（比如天文学、金融学、科技等），但数据科学却只是科技界的事儿？大数据，多大才是大？这些术语及概念如此含混不清，简直毫无意义。

(2) 对于数据科学领域的研究者，不管是在学术界还是工业界，公众都缺乏敬意。事实上，他们在这一领域内辛勤工作了很多年，而这些工作是继承了各个领域的前辈们数十年甚至数百年的工作成果，这些领域包括统计学、计算机科学、数学、工程学以及其他学科。而媒体传播给公众的信息却是这样的：机器学习算法是上个礼拜才发明出来的，谷歌出现之前都不存在所谓的大数据。这简直荒谬，很多正在使用的方法和技术，还有我们面临的挑战，都不过是在过去已有的方法、技术和挑战上演变而来的。我们并不否认新事物和新技术的出现，只是觉得应该对历史和前人的研究成果保持必要的敬意。

(3) 媒体疯了。人们将各种各样的桂冠加诸数据科学家的头上，人们形容他们是掌握了宇宙奥秘的魔法师，其疯狂程度堪比金融危机之前。天花乱坠的宣传很容易掩盖真相、歪曲事实。这些宣传的噪声越多，真正有效的信息就越少。因此，若“大数据”被媒体吹得越久，公众越容易被误导，越难获知这一概念背后真正有益于社会的一面（如果有的话）。

(4) 统计学家觉得他们正在干的事就是数据科学。换句话说，这本来就是他们的饭碗。亲爱的读者们，请设身处地替统计学家们想想，有人抢自己的饭碗是什么感受。媒体也常常将数据科学轻描淡写为统计学和机器学习在科技界的简单应用。我们会在书中阐明，不是说将统计学和机器学习这些“旧酒”装进新瓶里，就叫作数据科学。它绝对有资格作为一个独立的学科存在。

(5) 所有自称为科学的都不是真正的科学。这句话或许有些道理，但不代表数据科学这一术语毫无意义，它代表的可能不是科学，而是某种技术。

1.2 冲出迷雾

Rachel 取得统计学博士学位到她在谷歌工作的这段经历，或许能帮我们解答一些疑惑，她说：

> 进入谷歌之后，我很快就意识到工作中用到的东西和我读统计学博士学位时学到的东西差别很大。并不是说我的统计学知识毫无用武之地，相反，我在学校学到的东西为我思考问题提供了一个框架，统计学的很多知识都为我的日常工作提供了坚实的理论和实践基础。
>
> 在谷歌工作期间，我发现必须掌握很多在学校没学到的东西，比如计算、编程、数据可视化技能和许多领域知识。这种经验既特殊又普遍，我拥有统计背景，因

> 此需要补充前面提到过的那些知识，而若换作一位计算机、社会学或者物理学背景的人，他们也需要根据自己的知识缺陷去补充相应的知识。每个人都拥有自己独特的知识结构，重要的是大家能够紧密合作，取长补短，组成一个团队去解决数据问题。

一般人对上述故事肯定会有这样一种想法：你走上工作岗位后就会发现，在学校学到的知识，远远不能满足实际工作的需要。因此，本书中教授的统计学知识与业界所应用的统计学方法，肯定也是不尽相同的。对此，我们有一些自己的看法。

- 为什么学校里的统计要和工业界的统计如此不同？为什么很多学校的课程要和现实如此脱节？
- 这种差异不仅存在于学校里的统计和工业界的统计之间。很多数据科学家的一个共同感受是，工作时他们需要接触更多的知识、方法论和工序（详见第 2 章），而这些东西都是以统计学和计算机科学为基础的。

抛却这些媒体给予数据科学的光环，只有一件事是实在的：数据科学是一个新生事物。它刚刚诞生，却被赋予了太多荣耀，使人们对其充满了很多不切实际的幻想，而幻想最终是会破灭的。我们要保护数据科学，过分吹捧可能会让这个新兴领域过早夭折。

Rachel 决定去研究数据科学这一文化现象，她想了解其他人对数据科学的感受。她开始和谷歌的人接触，和很多创业公司和高科技公司的人接触，和大学（特别是统计系）里的老师们接触。

从这些接触中，Rachel 觉得数据科学的轮廓渐渐清晰起来，她进一步深入，决定在哥伦比亚大学开设一门数据科学导论课程，与此同时 Cathy 在博客上连载了该课程的讲义。我们期望在这门课程结束时，我们，还有我们的学生们能对数据科学的本质有一个清晰的理解。现在我们把课程的内容集结成书，也是希望帮助更多的人去了解数据科学。

1.3 为什么是现在

现在，数据充斥在我们生活的方方面面。网络购物、网上通信、浏览新闻、收听在线音乐、搜索信息，或在网上表达观点，这些行为都会被记录。同时，我们拥有充足且廉价的计算能力。有数据，有计算能力，这为从事数据科学提供了良好的环境。

大家都知道，线上数据的收集正在经历一场革命（稍后会详细介绍），但他们所不知道的是，离线数据的采集同样也在革新。人们的日常行为也被“数据化”了。将二者结合起来，我们可以深入研究人类的行为，甚至从更高的物种角度，来研究人类行为区别于其他物种的特殊性。

数据也不局限于互联网产生的数据，金融、医疗、制药、生物信息、公共福利、政府、教

育、零售等行业都会产生大量的数据，数据在各行各业的影响力在与日俱增。部分行业所储存的信息达到了“大数据”的程度（详见第 2 章），而另一些行业的信息量则没有那么多。

数据科学这一课题变得日益有趣（或提出了新的挑战），这不仅仅是因为数据的体量增大，更多的是因为数据本身（很多时候是实时数据）成了构建数据产品的关键要素。在互联网上，有亚马逊的推荐系统、Facebook 的朋友推荐系统，还有其他的图书、电影、音乐等推荐系统；在金融业，有信用评级系统、交易算法和模型；在教育领域，可以实现教育对学生的量身定制，比如现在的网络培训公司 Knewton 和网络大学 Khan Academy；在政府机构中，这意味着以数据为基础去制定公共政策。

我们正在见证一个时代的开始，这个时代是一个巨大的、充斥着人文特色的反馈环：我们的行为会改变产品，产品又反过来影响我们的行为。技术使这一切成为可能，我们拥有处理大数据的基础架构、更大的内存和更快的网络，而且社会公众也日渐认同技术是生活中必不可少的组成部分。在十年前，这一切我们还不敢想象。

由于这种基于反馈的循环对社会变革将产生不可小觑的影响力，我们认为，有必要认真考虑如何确保这种循环的良性运行，尤其是直接参与这一过程的人员，在实践中应保持哪些道德准则、应负何种责任。本书的目的之一就是针对这些话题开展一些抛砖引玉的探讨。

数据化

Foreign Affairs 杂志在 2013 年 5/6 月期刊上发表了一篇由库克耶和迈尔 – 舍恩伯格共同撰写的文章“The Rise of Big Data”（大数据的崛起）。该文谈到了数据化的概念，以朋友之间的关系为例，他们将对朋友的喜欢程度转化为数值，这些数据被存储起来，用于日后研究，或者出售。将问题数据化，这是我们人类处理问题时经常采用的一种方式，不管是线上还是线下。

在文章中，数据化被定义为一种处理流程，它将生活的方方面面转化为数据。比如，谷歌眼镜将其所视范围内的景象转化成数据，Twitter 将人们偶尔产生的想法转化成数据，LinkedIn 将职业社交网络转化成数据。

数据化是一个很有趣的概念，我们在重视它的同时，必须尊重他人的意愿——是否自愿与人们分享自己的数据。比如，在网上为某个人或某件东西“点赞”时，人们要么是故意让自己的行为“被数据化”，要么最低限度上也清楚自己的行为会被记录下来。但有时却不然，我们只是随意浏览一些网站，我们的行为却被网站上的 cookie 记录下来；我们走进商店，或者只是走在大街上，会被各种传感器、摄像头监测，或者被谷歌眼镜拍摄，我们的行为被作为数据存储下来，而这种数据化并非出于我们的意愿。

数据化无所不在，从作为实验对象参与到社交媒体实验中，到接受全面调查，再到被人秘

密跟踪，这些都是被数据化的典型案例，它们代表了数据化过程中个人意愿从高到低的各种情形，但其产生的结果却远不能如此简单地划分概括。

在文章中他们又说：

> "一旦我们可将问题数据化，就能改变人们的意图，并在这些信息基础上产生新价值。"

本书会不时提出这样一个问题：究竟谁才算"我们"？"新的价值"是什么？在他们的文章中，"我们"显然指那些模型和企业，他们引导用户购买更多的产品，赚取更多的钱，"新价值"则指那些能提高效率的方法，比如通过自动化等。

如果将视野放得更大，将这里的"我们"指代更广泛的人类，那就有点逆潮流而行的意思了。在面对数据化的大潮时，我们或许会有所保留。

1.4　数据科学的现状和历史

那么，到底什么是数据科学？它是一门新生事物，还是统计学的旧瓶子里装了新酒？它是纯粹的炒作，还是确有其事？如果它是一门实实在在的新兴学科，它的意义何在？

让我们先上网看看业界关于这一问题的讨论，这不一定能直接回答我们的问题，但听听别人怎么说总是有益的。2010 年，Quora 网站有一个关于"什么是数据科学"的提问，Metamarket 公司的 CEO Mike Driscoll 的回答如下：

> 研究数据科学，一方面需要如极客那般刻苦钻研，一方面需要像统计学家那样拥有完美的理论。
>
> 数据科学家不仅仅是极客——极客只关心如何调试一行 Bash 脚本或 Pig 脚本，没人会在意非欧氏距离矩阵。
>
> 数据科学家也不仅仅是统计学家——后者只关注如何完成一个理论的证明或构建出一个完美的模型，很少有人会使用 R 语言将数据文件读入系统，从而进行后续的分析。
>
> 数据科学是一门关于数据的工程，它需要同时具备理论基础和工程经验，需要掌握各种工具的用法。

Driscoll 随后引用了 2010 年 Drew Conway 的韦恩图来说明研究数据科学需要的技能，如图 1-1 所示。

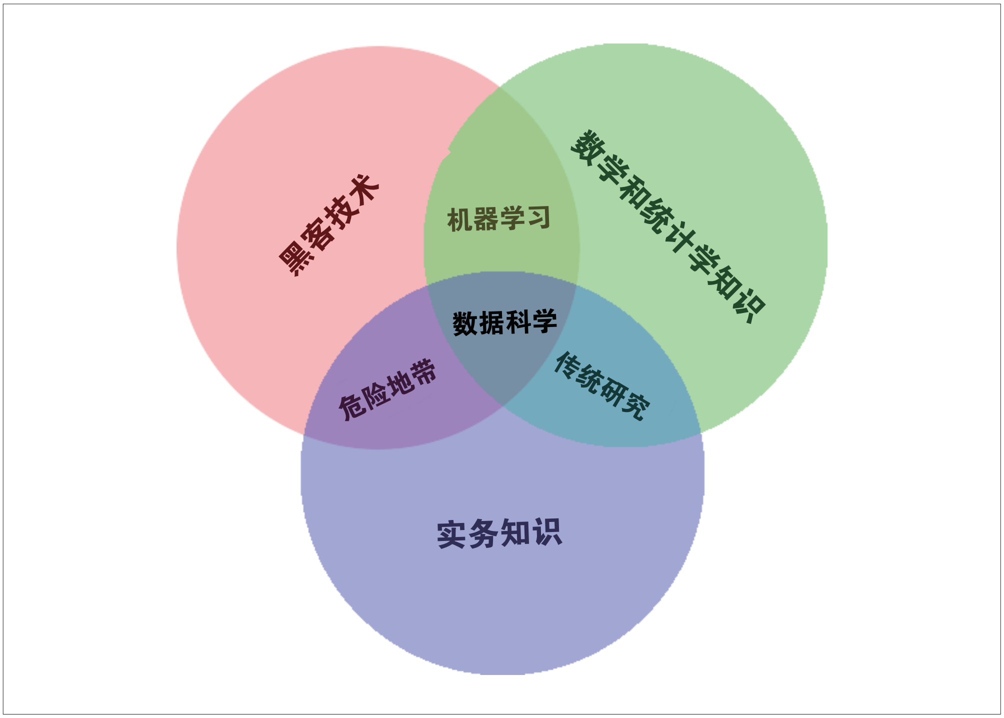


图 1-1：Drew Conway 的数据科学韦恩图

Driscoll 还引用了 Nathan Yau 2009 年的一个关于“数据科学家正在涌现”（Rise of the Data Scientist）的帖子，其中介绍了数据科学家们应该具备的各种技能：

- 统计学（做传统分析时需要的技能）；
- 数据处理（解析、提取和格式化数据）；
- 可视化（图表、工具等）。

但是先别忙，如此说来，数据科学就是这些技术的一个简单的组合吗？抑或是诸如统计学、机器学习等学科的一个逻辑上的扩展？

Cosma Shalizi[1] 和 Cathy[2] 分别就统计学家和数据科学家的区别这一问题发表了很多看法。Cosma 认为，任何一个够格的统计部门都在从事数据科学的工作，数据科学只不过是统计学换了个新说法。

持此观点的还有 ASA 主席 Nancy Geller，她在 2011 年发表的一篇文章“Don’t shun the ‘S’ word”中说：

注 1：http://goo.gl/SO7ceN 和 http://goo.gl/pXg1fU。

注 2：http://goo.gl/F4K4hE 和 http://goo.gl/X9Bmxj。

> 我们要告诉人们：是统计学家揭示出数据的含义。在21世纪，各行各业都涌现出了海量的数据，无论是科学、工程还是医学，从文学史到动物学，人们在处理这些数据时都应用了统计学技术。这种数据大爆炸，为统计学者提出了源源不断的研究课题，因此在这个时代从事统计学工作是一件相当令人兴奋的事。

Nancy 以为用“从文学史（Art history）到动物学（Zoology）”这种说法，就可以巧妙地暗喻“从头到尾”[3] 的概念，代表了数据科学的应用无处不在。但她这种说法却是搬起石头砸了自己的脚，因为她所罗列的全是学术界的例子，恰恰不包含高新技术企业，而业界才是数据爆炸式增长最迅猛的地方，数据科学也是在这些高新技术企业里得到了长足的发展。在企业中，会有数据科学家的职位，但这一称号在学术界还很少见到（或许这点会慢慢改变）。

不久前 DJ Patil 和 Jeff Hammerbacher 讲述了 2008 年，他们是如何分别在 LinkedIn 和 Facebook 上定义了“数据科学家”这一称谓的。2008 年，“数据科学家”成为一个职位，出现在这两家公司的招聘信息里（维基百科于 2012 年增加了数据科学的相关词条）。

当一组技术在谷歌得到追捧，而且这种势头蔓延到硅谷的其他高科技公司时，一个新的职位就会出现，而当这成为常态，人们就需要给它一个全新的名字，比如数据科学家。当这个新名字声名远播，所有人都希望自己成为一名数据科学家。《哈佛商业评论》（*Harvard Business Review*）把数据科学家誉为“21 世纪最性感的工作”，这无疑是火上浇油。

社会学家在数据科学中的角色

LinkedIn 和 Facebook 都是做社交网络的公司，他们所谓的数据科学家经常是对统计学家、软件工程师和社会学家的统称。这很好理解，因为他们的产品就是社交工具，主要处理的内容是个人（用户）行为。但是根据 Drew Conway 的韦恩图，数据科学所研究的问题经常是跨领域的，也就是需要大量的“实务知识”（见图 1-1）。

也就是说，数据科学家要用到哪些“实务知识”，就要具体问题具体分析了。如果你要解决的是跟社交网络相关的问题，比如说“好友推荐”“可能认识的人”以及“用户分类”等，那一定要把社会学家拉进来。社会学家大多都擅于提问，他们也热爱调查研究，如果他们再会定量分析和编程，肯定会成为优秀的数据科学家。

由于“历史”的原因（其实不过是 2008 年的事），人们认为数据科学家的工作只是负责分析在线用户的行为数据。而现在兴起了一个全新研究领域，它被称作“计算社会科学”，我们可以将其视作数据科学的一个子集。

让我们回到更早的 2001 年，当时 William Cleveland 写了一篇关于数据科学的文章“Data Science：An action plan to expand the field of statistics”。

注 3：from a to z，意即“完全、彻底”。

那么，是先有的数据科学还是先有的数据科学家？

这就引出了一系列问题：我们能通过数据科学家的工作来定义数据科学吗？谁有资格定义这个全新的学科？媒体制造了很多关于数据科学的时髦用语，但他们有资格定义吗？我们需要依赖于这些自诩的数据科学家吗？到底有没有这样一个权威机构？我们暂且不予回答。

数据科学的职位

在布隆伯格的帮助下，哥伦比亚大学决定成立一个新的研究所用于数据科学和工程方面的研究。据我们上一次统计，仅在纽约就有 465 个数据科学的就业机会。即使数据科学还算不上一个真正的领域，但它已经在产生实实在在的工作职位了。

在这些招聘职位的描述中我们发现，数据科学家被要求具备计算机科学、统计学、传播学、数据可视化等领域的知识，还要是一个“通才”。但事实上，没有人能如此面面俱到，因此组建一个具备多种技能的团队更为可行。通过组建团队让不同领域的专家通力合作，这基本上就可以达到数据科学家的“通才”的要求了。我们先来看看现今的数据科学家需要具备哪些素质。

1.5　数据科学的知识结构

在数据科学导论的课堂上，Rachel 发给每个学生一张卡片，让他们根据在如下领域的技能水平填写自己的知识结构：

- 计算机科学
- 数学
- 统计学
- 机器学习
- 某一领域的专业知识
- 沟通和演讲的技巧
- 数据可视化

图 1-2 显示了 Rachel 在数据科学方面的知识结构。

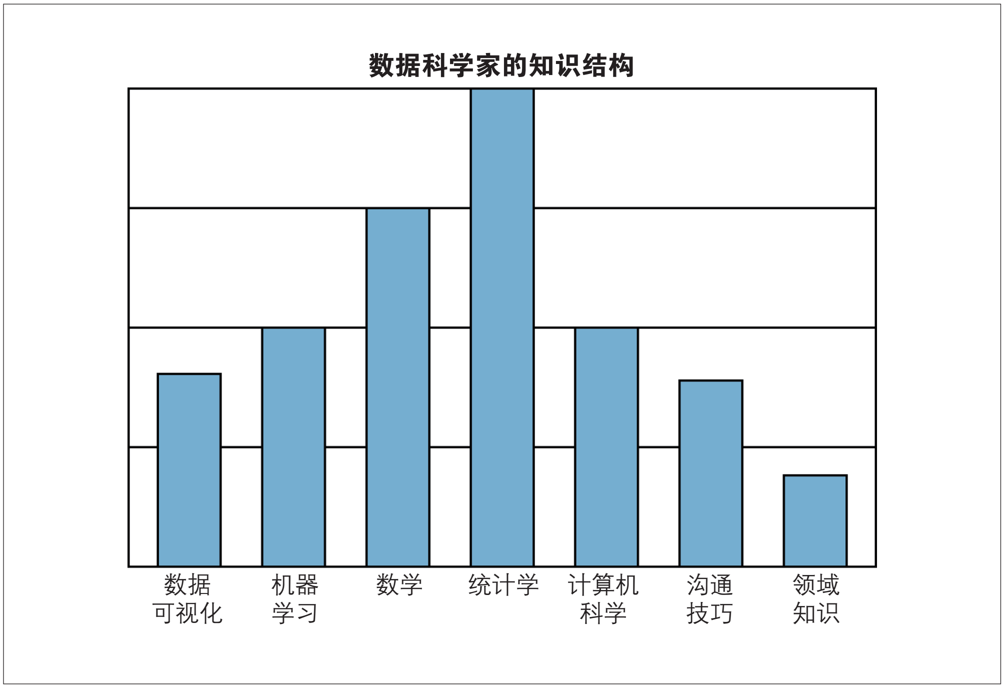


图 1-2：Rachel 的数据科学知识结构，她试图以此图描绘一个数据科学家应该具备的技能。她希望学生们和客座讲师们都能绘制自己的图谱，并且通过这样的自我检视来发现知识结构中存在的不足

我们把这些卡片钉在黑板上审视一番，发现个体之间技能上的差异还是很大的，这点让我们很满意。比如说，学生中很多都拥有社会学的教育背景。

你在数据科学方面的知识结构是什么样子的呢？你想它在几个月后变成什么样呢？几年后呢？

像我们早先提到的那样，最佳选择可能就是让拥有不同技能的人组成团队进行数据科学方面的工作，因为没人可以掌握所有的知识。于是，我们开始思考，相较于定义“数据科学家”，是否定义“数据科学团队”更有意义？图 1-3 定义了一个数据科学团队。

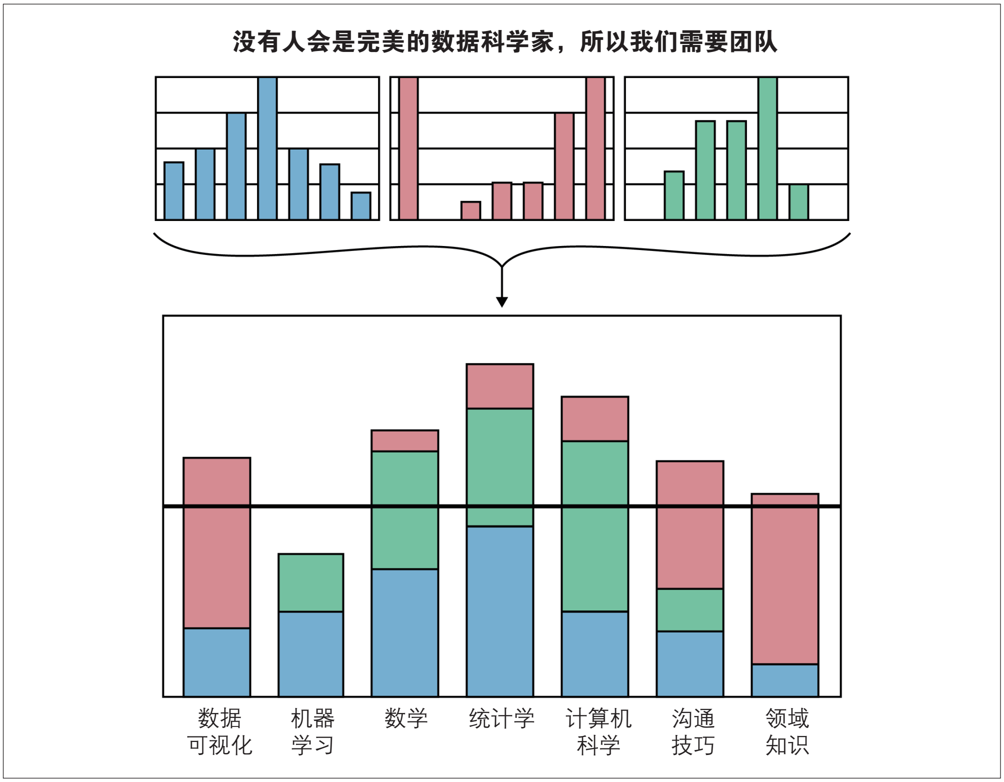


图 1-3：数据科学团队的知识结构由每个成员的知识结构叠加而来，在组建团队时，要让团队技能与所解决的问题大致匹配（另见彩插图 1-3）

1.6 思维实验：元定义

每一节课上都会有一个“思维实验”的环节，我们把学生分成小组来讨论问题。很多问题都是开放性的，我们只想借此引发学生就数据科学的相关问题展开更广泛的讨论。在第一节课上，我们的思维实验是：可以通过数据科学的手段来定义数据科学吗？

通过分组讨论，同学们提出了一些有意思的想法。

- *使用文本挖掘模型*
  首先在谷歌上搜索“data science”（数据科学），对搜索结果进行文本挖掘。但在语言的选用上，*使用者*和*从业者*的原则是截然不同的。作为使用者，我们会采用大众的定义（所谓大众的定义，即通过谷歌搜索得来的结果）。而对于从业者而言，若能引用权威渠道（比如《牛津英语词典》）的说法来定义数据科学会更严谨一些，但可惜的是这些词汇恐怕尚未收入其中，而且我们也没有耐心去等待了。所以，我们不得不承认，数据科学的定义包罗万象，但目前没有一种定义能让各方都满意。

- 使用聚类算法

  何不考虑数据科学的从业者，看看他们是怎么形容自己的工作的（也许最开始是“单词云”的形式）？然后，我们再看看其他行业的从业者，比如统计学家、物理学家、经济学家，看看他们又是怎么形容自己的工作的。然后，我们使用聚类算法（将在第 3 章用到）或者其他模型，看看根据对工作内容的描述，是否可以预测出其从事的行业。

作为对比，让我们来看看 Harlan Harris 是如何进行调查，使用聚类算法定义数据科学的子领域的，如图 1-4 所示：

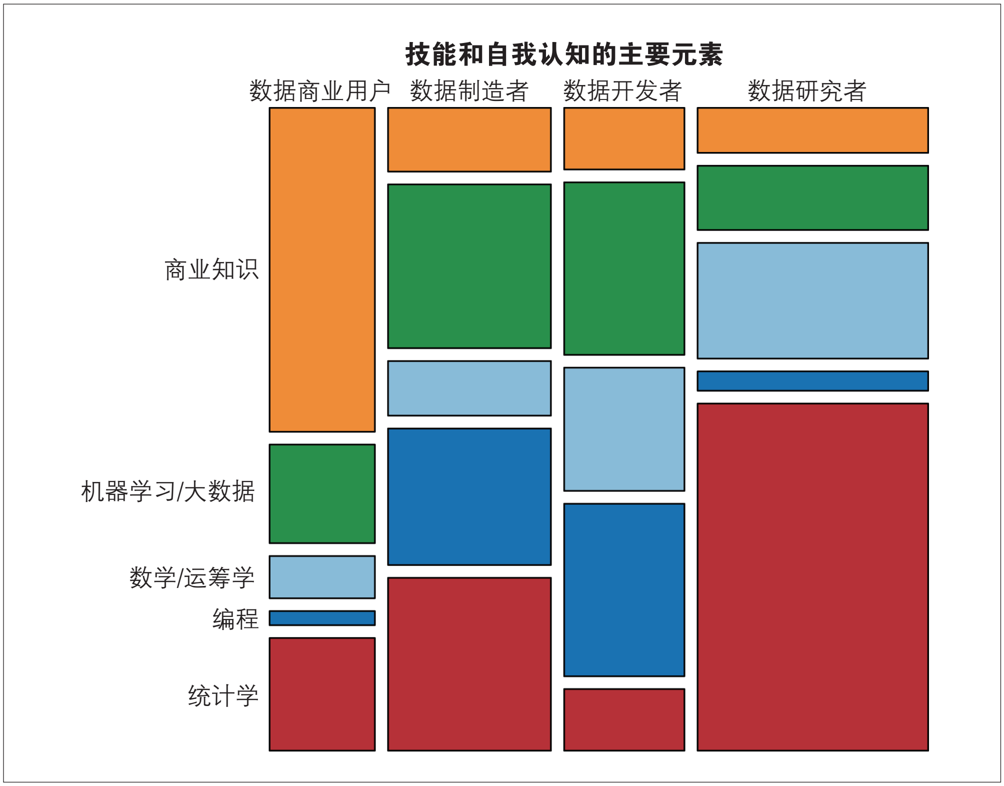


图 1-4：此图使用聚类算法描述了数据科学的子领域，源自 Harlan Harris、Sean Murphy 和 Marck Vaisman 基于 2012 年年中对数百名数据科学从业者的调查所著的 *Analyzing the Analyzers*（O'Reilly）（另见彩插图 1-4）

1.7　什么是数据科学家

也许定义数据科学最具体的方式是看它如何被使用，比如雇主们都花钱让数据科学家去做哪些工作。以此为目的，我们将会具体说说数据科学家究竟都在干什么，不过我们先来看看学术界。

1.7.1 学术界对数据科学家的定义

在学术界，现在还没人称自己是数据科学家，除非他们工作于某大学的“数据科学研究所”，或者在申请数据科学研究的经费，这时，他们才勉强将数据科学家作为自己的第二称谓。

不如我们问另外一个问题：在学术界，哪些人打算成为数据科学家？在哥伦比亚大学的数据科学导论课学习的有 60 名学生，Rachel 打算开设此课时，估计这门课的学生主要来自统计学系、应用数学系和计算机科学系。事实上，后来学生背景的多样化大大超出了她的预想：除过上述三个领域之外，她的学生还有来自社会学、新闻学、政治学、生物医学信息学、建筑学、环境工程、纯数学和商业学院的，此外还有来自纽约市政府机构以及关注社会福利的非盈利性机构人员。其中不乏一些已经在从事数据科学工作的人。他们都很希望能使用数据去解决一些重要的问题，通常这些重要问题具有重要的社会价值。

想要使“数据科学”在学术界立得住脚，其所要研究的领域应该有更规范的定义。值得一提的是，现在数据科学领域已经有很多可以转化成博士论文的研究课题。

让我们试着定义数据科学家：一个学术界的数据科学家首先是个科学家，他接受了任何其他学科的训练（从社会学到生物学等各种学科），还要同大量的数据打交道，不管这些数据的结构、规模以及复杂程度如何，他都能挖掘出数据背后的意义，从而解决现实世界中的问题。

上述例子说明，在不同的学术领域人们面临的计算和深度数据问题都存在较大的共性。若不同机构的研究者通力合作，他们就可以解决各个领域的现实问题。

1.7.2 工业界对数据科学家的定义

那么工业界的数据科学家又在做些什么？这取决于数据科学家的资深程度以及是否将数据科学特别限定在互联网领域。数据科学家这个职位也不是只有科技界才有，但是数据科学这个词的确源自科技界，为了避免混淆，我们就将业界特指为科技界。

首席数据科学家将为公司设定数据策略，这包括筹建用于收集数据和记录日志的基础架构，确定如何在收集数据的同时保护隐私、哪些数据是面向用户的、如何使用数据做决策，又如何把这些数据反过来应用于产品设计，提升产品质量。他要管理一个由工程师、科学家和分析师组成的团队，还要负责和公司的管理层（如 CEO、CTO 等）进行沟通。他还负责为创新性的成果申请专利和设定研究目标。

更广泛地看，数据科学家是这样一种人，他懂得如何从数据中抽取信息并且解释数据背后的意义，这需要掌握统计学和机器学习中的工具和方法，还要具备人文主义精神。他要花费大量时间来采集、清理和处理数据，因为数据永远都不会是整齐规范到让人一眼可以读

懂的。在这个过程中，他需要坚持不懈，需要统计学和软件工程的技巧，而这些也是理解数据的偏差、调试程序时所必备的技能。

当数据被整理成型后，他需要结合可视化和数据的意义对数据进行探索分析。他会找出模式，构建模型，设计算法——有些是为了了解产品的使用情况和整体质量，有些是为了搭建原型，将在这些原型上经过验证的东西重新揉入产品中，从而提升产品品质。他会设计实验，他是基于数据做出决策这一过程中的关键一环。他要使用明白无误的语言和图形同组内成员、工程师、领导层交流，即使有人对数据不是很敏感，也可以通过他知道这些数据背后的意义。

这就是对数据科学的一个概览，本书将帮助你了解其中大多数内容。接下来，让我们停止空谈，开始进入数据科学的实战阶段！

第2章

统计推断、探索性数据分析和数据科学工作流程

在本章，我们首先讨论统计推断和统计学的思考方式，然后我们将目光转向每一位数据科学家都会从事的工作：探索性数据分析。我们还将详细了解研究数据科学的工作流程，在本章的结尾，是我们的“思维实验”环节和一个案例学习。

2.1　大数据时代的统计学思考

> “大数据这个词现在时常被人们随意使用，然而其语义十分模糊。简单地说，这个包罗万象的词条一般有三层含义：首先，它指代一揽子的技术；其次，它有可能引发一场度量数据规模的革命；最后，它为人们未来将会、甚或是应该如何制定决策提供了一个新视角，一种新理念。”
>
> ——史蒂夫·洛尔（Steve Lohr）
>
> 《纽约时报》

在你打算成为一名数据科学家时，以下技能是必须首先具备的：统计学、线性代数和一些编程技能。同时你还需要发展以下技能：数据预处理、数据再加工、数据建模、编码、可视化和有效沟通，这些技能往往是相辅相成的。这些内容会在本书中交叉出现，万丈高楼平地起，让我们先从统计推断开始探索数据科学的旅程。

作者希望本书的读者拥有五花八门的背景，你可能是位优秀的软件工程师，有能力搭建数

据管道，但对统计学却所知甚少；或者你是一位市场分析专员，一点也不懂如何编写程序；或者你只是一位对数据科学充满好奇的读者，想弄明白数据科学到底是什么。

虽然我们在这里列举了阅读本书需要具备的一些先决条件，但我们不是查户口的，无法到你家去检查你是否修习过有关统计学的课程，或者是否曾经看过有关统计学的书籍。即使你曾经选修过诸如统计学导论之类的入门课程，但正如我们在鸡尾酒会上经常听到的那样，99% 的人都十分惧怕统计学，宁愿从来没上过这门课。如此说来，你不大可能从这些课程中真正领略到了统计学的美妙，更不可能深入研究它。

即使你取得了统计学的博士学位，已经对这一领域有精深的研究，你也可以通过阅读本书回顾一些基础概念，记起什么是统计推断、什么是统计学的思考方式，这总是有所助益的，特别在“大数据”时代这个全新的语境下，很多传统的统计学方法可能都需要重新审视和修订。

2.1.1 统计推断

我们所处的世界异常复杂，充满了随机性和不确定性，同时，这个世界又是一个巨大的数据生产机器。

我们搭地铁或开车去工作，我们的血液在身体里流动，我们购物、收发电子邮件、因浏览网页或查看股价而耽误正事；我们工作、吃饭、和朋友家人聊天；工厂里生产产品……这些行为都会直接或间接产生数据。

试想花一天时间去观察窗外的人流，记录下每一个经过的人；或者聚集起住在你一公里以内的人，问他们在过去的一年中每天收到多少封电子邮件；或者去你附近的社区医院翻阅血液样本，去发现蕴藏在其中的 DNA 模式，这些行为乍一听觉得有点变态，让人觉得不安，但事实并非如此。我们的生活本身就是不断产生数据的过程。

我们想用多种方式去描述、揭示这些过程，使人们更好地理解数据的产生。作为科学家，我们想更好地了解这个世界。对过程的了解掌握，也是解决问题的一部分。

数据就是现实世界运转留下的痕迹。而这些痕迹会被如何展示出来，则取决于我们采用什么样的数据收集和样本采集方法。假如你是数据科学家，那么作为一个观察者，你要做的事是将具象的世界转化为抽象的数据，这个过程是绝对主观的，而非人们所想的那么中立客观。

将这一过程从数据收集中剥离出来，就能清晰地看到蕴藏其中的随机性和不确定性的两个源头：一是来自过程本身，二是来自数据采集方法。

从某种程度上说，掌握了数据就掌握了世界，或者世界的运作规律，但是这并不代表着，你拿着一个巨大的 Excel 文件，或者存有数百万条记录的数据库，手指轻轻一划，就能理

解世界及其运行规律。

你需要新的点子，将这些采集到的数据进行简化，使它们更易于理解，能够以一种更简明扼要的方式概述世界运行的规律，能够易于使用数学对其进行建模的数据，这称为统计估计量。

这一套从现实世界到数据，再由数据到现实世界的流程就是*统计推断*的领域。

更准确地说，统计推断这门学科主要关注如何从随机过程产生的数据中提取信息，它是流程、方法和理论的统一。

2.1.2 总体和样本

让我们先来统一一些术语和概念。

在经典统计学理论中，有*总体*和*样本*之分。英语中总体和人口是同一个单词，因此一说这个词，人们就会马上联想到：美国人口总数 3 亿、全世界人口总数 70 亿等。但是，在统计推断中，总体并不特指人口，它指的是一组特定的对象或单位，比如推特上发布的消息，照片或者天上的星星等。

如果我们可以度量和提取这些对象的某些特征，就称为对总体的一组*观察数据*，习惯上，使用 N 表示对总体的观察次数。

假设总体为去年“巨无霸”公司员工发送的所有电子邮件，则对该总体的一组观察数据可以包括以下内容：发信人的姓名、收信人列表、发信日期、邮件内容、邮件字数、邮件中的句子数、邮件中动词的个数、从邮件发出到获得第一次回复中间的时间等。

接下来就要采集样本。所谓*样本*，是指在总体中选取的一个子集，用 n 来表示。研究者记录下样本的观察数据，根据样本特征推断总体的情况。采样的方法多种多样，有些采样方法会存在偏差，使得样本失真，而不能被视为一个缩小版的总体，去推断总体的特征。当这种情况发生时，基于样本分析所推断出来的结论常常是失真甚或完全错误的。

在上述“巨无霸”公司的例子中，可以*随机*抽取全部员工的 1/10，以他们所发的邮件组成样本，或者*随机*选取一天，以该天内所发邮件的 1/10 作为样本。这两种抽样方式都是合理的，而且样本的数量也是相等的，但是，如果你据此计算每人发送电子邮件的数量，进而估计出“巨无霸”公司员工的发送邮件分布情况的话，应用两种采样方式，你将会得到完全不同的答案。

这样一个简单的问题，由于采样方式的不同，结果都会失真，那么对于那些复杂的算法或模型，如果你没有把获取数据的方式考虑进来，情况又会怎样？

2.1.3 大数据的总体和样本

在大数据时代，我们有能力记录用户的所有行为，我们难道不就可以观察一切？那么，此时做总体和样本的区分还有意义吗？如果我们已经拥有了所有的电子邮件，干嘛还要采样？

这些疑问直抵问题的核心，对此，我们有如下几个方面需要加以澄清。

- *采样可以解决一些工程上的挑战*

  在时下流行的关于大数据的讨论中，企业都主要采用 Hadoop 等分布式技术去解决海量数据带来的工程及计算问题，但他们却忽略了采样这种手段也同样有效。事实上，在谷歌，软件工程师、数据科学家和统计学家时刻都在用到采样来处理大数据。

使用多少数据取决于你的目标：比如，做分析或推断，你只需要部分的数据就可以了；但当你试图在用户界面上展示其中一个使用者的信息时，你可能需要搜集该使用者的所有数据。

- *偏差*

  即使我们拥有了谷歌、Facebook 或 Twitter 的所有数据，基于这些数据所做出的统计推断并不适用于其他不使用这些服务的人群，即使针对使用这些服务的人群，上述统计结论也不能准确说明他们某一天的活动轨迹。

微软研究院的 Kate Crawford 女士在她的演讲“Hidden Biases of Big Data”中提到，如果仅对飓风桑迪到来前后的推特做分析，可能会得出这样的结论：人们在飓风来临前在购物，飓风过后在聚会。然而，这些推特大部分来自纽约人。首先，他们是推特的重度用户，而这时沿海的新泽西人正在担心他们的房子会不会被飓风吹倒，他们哪里还有时间和心情去发推特呢？

换句话说，如果你使用推特数据来分析桑迪飓风的影响，得出的结论可能会是：这场飓风危害不大。事实上，你得出的结论只能说明飓风对推特用户的影响。他们受桑迪飓风的影响很小，还有时间来发推特，但他们无法代表一般意义上的美国大众。

同样在这个例子中，如果你不了解相关的语境或者对桑迪飓风一无所知，你就不可能对数据做出合理的解释。

- *采样*

  让我们再来看看总体和样本在各种语境下的含义。

在统计学中，我们通常使用一种基础的数学方法来建模，描述总体和样本的关系。针对背后可能存在的规律、数学结构以及生成数据的过程，我们会做出一些简单假设。每一次研究，我们对其中一种数据生成过程中所采集到的数据进行观察，这组数据就是所谓的样本。

以“巨无霸”公司的邮件为例，如果我们随机抽取阅读一些邮件，就会产生一个样本。但如果我们再次抽取，又会产生一组完全不同的样本。

由于采样过程不同所带来的不确定性有一个学名：取样分布。就像 2010 年上映的、由莱昂纳多 · 迪卡普里奥主演的《盗梦空间》一样，这是一个梦中梦。因此，也可以将“巨无霸”公司的电子邮件想象成一个样本，而不是总体。

这些电子邮件是一个更大的超级总体的样本（此刻，我们有点哲学家附体），如果抽样时是用扔硬币来决定的话，硬币若多翻转一次，得出的样本就会完全不同。

在这里，我们使用一组电子邮件作为样本，来推断其中一个数据产生的过程，比如说：“巨无霸”公司员工的电子邮件书写习惯。

- 新的数据类型

  过去，所谓数据是指数字和一些分类变量，但今非昔比。在大数据时代，一个优秀的数据科学家需要多才多艺，要处理的数据种类比过去要多得多，如下所示。

  - 传统数据：数字、分类变量和二进制变量。
  - 文字：电子邮件、推特、《纽约时报》上的文章（详见第 4 章和第 7 章）。
  - 记录：用户数据、带有时间戳的事件记录和 JSON 格式的日志文件。
  - 地理位置信息数据：将在本章使用纽约市的房屋数据为例做以简单说明。
  - 网络（详见第 10 章）。
  - 传感器数据（不在本书讨论范围内）。
  - 图片（不在本书讨论范围内）。

这些新的数据类型要求我们在做采样时需更谨慎。

以 Facebook 用户产生的实时数据流为例，从带时间戳的日志上，可以抓取用户一周内的活动数据，在此基础上进行分析，得出的结论适用于下周或者下一年吗？

在复杂的网络中你又如何采样，使得样本可以反映总体的复杂性？

这些问题都是统计学和计算机科学领域内的开放性研究问题，我们本来就身处科技的前沿。在实际中，数据科学家尽其所能去解决这些问题，在他们的工作中，经常发明出一些创新性的方法。

术语：大数据

我们已经多次提到了“大数据”，但是一直没有好好定义它，只是在上一章中围绕它提出了一些问题，我们也觉得不好意思，那就让我们先来看看什么是大数据。

大数据的大是相对的。人为地为大数据限定一个阈值，比如 1PB，是没有意义的，这太绝对了。只有当数据的规模大到对现有技术（比如内存、外存、复杂程度、处理速度等）构成挑战时，才配称为“大”。因此，大数据的大是一个相对概念，大数据放在 20 世纪 70 年代和现在的意义是完全不同的。

当用一台机器无法处理时，就可以称为“大数据”。不同的人、不同的公司，拥有的计算资源是有差别的，对于数据科学家来说，如果数据大到一台机器处理不了，就可以称其为“大数据”，因为她不得不学习使用一些全新的工具和方法去解决这一问题。

“大数据”是一种文化现象。它描述了数据在人类生活中所占的比重，随着科技的发展，数据所占的比重越来越大。

大数据中的 4V 原则。这 4V 是指容量（Volume）、种类（Variety）、速度（Velocity）和价值（Value）。很多人借此来描述大数据的特征，你也可以从中学习借鉴。

2.1.4 大数据意味着大胆的假设

还记得在第 1 章，我们提到了库克耶和迈尔 – 舍恩伯格的文章“The Rise of Big Data”，在文章中，他们提出大数据的革命由以下三方面构成：

- 采集和使用大量的数据，而不是小样本；
- 接受数据中存在杂乱噪声；
- 重视结论，放弃探究产生结果的原因。

他们将这三方面说得冠冕堂皇，他们宣称，数据是如此巨大，没有必要去寻找原因。也不用担心采样出错，因为所有的数据都在这，它们记录了一切事实。之所以这样说，是因为他们声称找到了处理大数据的新方式，那就是让“N = 全部”。

N 能代表全部吗？

答案是 N 永远不能代表全部。我们经常忽略那些我们最应该关心的事实。

以 InfoWorld 上的一篇帖子为例，互联网监控永远也不可能奏效，我们最想抓住的那些罪犯，恰恰是非常聪明、精通技术的，他们永远领先一步，永远也不会被人抓住。

那篇文章举了一个选举之夜投票的例子，这个例子中本身就存在矛盾之处：即使我们能把所有离开投票站的人都纳入统计，我们也还是遗漏了那些从一开始就不打算投票的人，他们那天晚上压根儿就没来投票站。而这些人或许才是我们需要关注的人群，和他们交谈才能了解我们国家现有投票制度存在的问题。

事实上，我们认为“N = 全部”这个假设是大数据时代人们面临的最大问题。首先，这种假设天然地将一大部分人排除在外，他们可能是因没有时间、精力、渠道去参加那些非正式或者未经宣布的选举投票。

在统计选票时，我们忽略了那些同时打两份工的人，还有那些把时间都花在等公交车上的人，他们因为各种原因没有投票。对你来说，这或许只意味着 Netflix 的推荐引擎推荐给你的电影不符合你的口味，因为在 Netflix 上愿意费心去为电影打分的大多是年轻人，他们的口味可能和你的不一样，这些打分行为使得推荐引擎更贴近他们的喜好。但是，上述例子中提出的基本观点在现实生活中很可能会埋下许多潜在隐患。

数据是不客观的

若说“N = 全部”这一假设可以成立，则意味着要承认数据是客观的。但是，相信数据是客观的，或者“让数据说话”，这些观点是错误的，当然，对于那些持完全相反观点的人也应该小心提防。

最近，我们在《纽约时报》上发表了一篇关于运用大数据方法来招聘人员的文章（http://nyti.ms/18OEq6j），正是这篇文章使我们意识到上述观点的可怕。文章中引述了一位数据科学家的说法：“让我们把所有东西都放进来，让数据自己说话。”

通读全文后，你会认识到，运用大数据运算法则是为了寻找到有潜质的“璞玉”式人才。这种做法在招聘中值得一试，但是有些事你要深思熟虑。

假设你面前有两位资历相当的求职者，一位是男性，一位是女性。阅读他们的简历你发现，相比之下，这位女性求职者跳槽次数更多，获得提拔的机会较少，而且对以前工作过的地方有更多的负面评价。

当再有这样的情况出现时，你若通过模型做决策，可能会雇用男性而不是女性求职者。你的模型是不会把有些公司歧视女性的情况考虑进去的。

换句话说，忽视因果关系是大数据法则的一种缺陷，而不是特征。忽视因果关系的模型无助于解决现存问题，而只会增加更多问题（在第 11 章将会予以详细说明）。数据也不会自己说话，它只能够以一种量化的、无力的方式去描述、再现我们身边的社会事件。

$n = 1$

与“N = 全部”这一极端观点相反的是另一个极端：$n = 1$，意思是说样本的总数为 1。过去，说样本空间的大小为 1 是很荒谬的，没人会通过观察一个个体，就得出对总体的推断。别担心，这种观点现在仍然是荒谬的。不过，在大数据时代，$n = 1$ 有了新的含义，对于一个人，我们可以记录所有关于他的信息，我们可以对所有举动进行采样（比如他打过的电话、他敲击键盘的记录），并对这些行为进行统计推断。这是一种用户级别的建模。

2.1.5 建模

下一章将会讲述对于采集到的数据如何进行建模，本章，我们先讨论什么叫建模。

Rachel 曾经给一位朋友打电话，讨论建模交流会的事，几分钟后她意识到，“model”这个单词对他俩来说完全是不同的含义。对方理解成了数据模型（data model），一种存储数据的结构，这属于数据库管理员的研究领域。而 Rachel 的意思是统计学中的模型，这也是本书大部分时间要讨论的内容。更可笑的是，最近 Andrew Gelman 的一篇关于建模的博客被时尚圈的人士发到了推特上，他们可能理解成了模特。

即使你已经使用*统计模型*或*数学模型*这两个术语很多年了，但当你向周围人谈起模型时，你和他们都明确知道这个词的含义吗？什么使一个模型成其为*模型*？当我们问起这些基本问题时，还有一个问题也很重要，统计模型和机器学习算法之间有什么不同？

在深入这些问题之前，让我们先来看看 Chris Anderson 2008 年在《连线》杂志上发表的文章“The End of Theory: The Data Deluge Makes the Scientific Method Obsolete”，这篇引人争议的文章为我们的讨论增加了更多素材。值得一提的是，Chris Anderson 那时候是《连线》杂志的主编。

Anderson 认为数据即信息，并且宣称不需要模型，了解相关性就够了。以海量数据为例，“谷歌根本没有必要使用模型”。

是这样吗？我不相信，读完此书，我觉得你们也不会相信。这种观点类似库克耶和迈尔－舍恩伯格在他们的文章中提出的“N = 全部”，我们刚刚在前面讨论过。现在，你也许已经可以感受到环绕在我们周围的深深的疑惑了吧。

然而我们需要感谢媒体，是他们让公众了解了这些问题，可是，当这些意见领袖们并不是专职从事数据科学工作的人员时，对他们的说法就得打一个问号了。仔细想想你是否同意 Anderson 的观点，哪些部分你认为是对的，哪些地方你认为是错的，或者你需要获取更多信息才能形成自己的观点。

鉴于公众对数据科学和模型的认知都来自主流媒体这样业余的描述，作为数据科学家，我们应该义不容辞地发出自己的声音，贡献出自己的真知灼见。

在这个大背景之下，当我们谈起*模型*时，其含义究竟是什么？数据科学家是如何使用它们的？让我们直接进入下一节，来回答这些问题。

什么是模型？

人类试图用各种方式去描述他们所处的世界。建筑学家用蓝图和三维立体模型来捕捉建筑的属性；分子生物学家用连接氨基酸的三维图像描述蛋白质结构；统计学家和数据科学家则用函数表示产生数据的过程中存在的不确定性和随机性，并以此来形容数据本身的样貌

和结构。

模型就好像一个特殊的镜片，我们透过这个镜片去观察和了解现实世界的本质，这个“镜片”可能是建筑学、生物学或数学模型。

模型是人工设计的，用于将无关紧要的细节排除或抽象化。在进行模型分析时，研究者必须关注这些被省略的细节。

以蛋白质为例，一种本身带有侧链的蛋白质骨架却不受规范电子运行轨迹的量子力学理论的约束，最终决定了蛋白质的构架和行为。以统计模型为例，建模时我们可能错误地排除了一些关键变量，而使用了一些与问题无关的变量，或者采用的数学结构偏离了问题本身。

统计建模

在引入数据和开始编写代码前，对于建模的流程有一个大概的了解是有益的。先干什么？谁受谁的影响？什么是因，什么是果？检验结果如何？这些都是我们应该思考的问题。

人们使用的方法各有不同，有些人喜欢用数学去描述这种关系。通用的数学公式里面必须包括参数，但是参数的值是未知的。

按照惯例，数学公式里一般使用希腊字母表示参数，拉丁字母表示数据。比如你有两列数据：x 和 y，你认为二者之间是线性关系，用公式表达如下：$y = \beta_0 + \beta_1 x$，此时你还不知道 β_0 和 β_1 的具体数值，因此，它们就是参数。

还有些人则喜欢画图。他们先画一张数据流的图，很可能带有箭头，用来描述事物之间是怎么相互影响的，或者在一段时间内发生了些什么。在选择公式表达这种关系之前，这种关系图可以给他们一个大概的描述。

如何构建模型

你怎么知道什么数据该用什么模型？这一半是艺术，一半是科学。这个问题正是打开数据科学大门的钥匙，可惜的是，本书中就这个问题能够给出的指引非常有限。只能说模型的选择是建模过程中的一环，你需要对底层结构做出大量假设，应该有一个标准来规范如何选择模型和解释这样选择的理由。但是我们还没有统一的规范，所以只能摸着石头过河，希望经过深思熟虑，能制定这样一套规范。

必须承认，我们也不知道从哪儿开始，如果知道的话，我们已经知道了生命的意义。但是，我们会尽力，尽力在书中向你展示我们在面对这样的问题时要怎么做。

也许探索性数据分析（EDA）是一个好的开始，我们将在下面一节介绍它。它牵扯到绘制图形和从数据集中获取直观的感觉。与试错、反复实验一样，探索性数据分析对问题的解决大有帮助。

实话实说，除非你做过很多次，否则这一过程在你眼里依然是那么神秘。最好是从易到难，先做看起来最傻的事，事后看，或许没有你想象得那么傻。

举例来说，你可以（或者说是应该）先绘制直方图或散点图以对数据产生一个直观的感受，然后试着写点什么，哪怕一开始是错误的（很可能一开始的结论是错误的，没关系）。

比如写出来的是一个线性方程（详见下一章），当你把它写下来，就会强迫自己去思考：这个方程有意义吗？如果没有，为什么没有？那怎样的方程对这个数据集是有意义的？你从最简单的方式开始，逐渐增加复杂度，做出假设并把你的假设写下来。如果你觉得完整的陈述有帮助，就把句子写完整，比如："我假设将用户自然分成五组，因为销售代表谈起用户的时候，她把用户分成了五类。"然后试着用方程式和代码来表达这一陈述。

记着，从简单处着手永远是个好办法，建模时在简单和准确之间有一个权衡。简单的模型易于理解，很多时候，原始简单的模型帮你完成了 90% 的任务，而且构建该模型只需要几个小时，采用复杂的模型或许会花上几个月，而且只将这个数值提到了 92%。

在本书中，你将从构建模型开始，它们将组成你的"兵工厂"。在构建模型时会用到很多模块，其中一种就是*概率分布*。

概率分布

概率分布是统计模型的基础。在讲述线性回归和朴素贝叶斯时，你就会知道我们这么说的原因。概率论是一门需要花费几学期去教授的课程，将这样庞杂的内容压缩并在一节中讲述，对我们来说实在是个巨大的挑战。

回到还没有发明计算机的年代，科学家观察到了现实生活中的一些现象，经过测量后发现，一些固定的数学模式在重复出现。其中的经典案例莫过于人的身高服从正态分布，正态分布是一种钟形曲线，也叫高斯分布，以数学家高斯的名字命名。

还有其他一些经常出现的曲线，也分别以各自的发现者的名字命名（比如泊松分布、韦伯分布等），另外一些曲线，如伽玛分布、指数分布，则是以描述它们的数学方程式命名。

自然状态下产生的数据，可以用数学函数来描述。通过设定函数中的参数，可使函数曲线接近于实际数据的分布形态。而这些参数可在对数据进行估计的基础上得出。

不是所有过程产生的数据都服从某种已知的分布，但很多都服从。我们可以应用这些分布函数作为模块，来构建最终的模型。本书兵不打算深入探讨每种分布的细节，但我们提供了图 2-1 用来展示这些常用的分布，事实上有无穷多种分布，列在这里的这些只是因为有人观察了它们很久，觉得有必要给予命名而已。

概率分布可以理解为对于可能结果的子集指定一个概率，概率分布用与其对应的函数来表示。比如正态分布的函数为：

$$N(x \mid \mu, \sigma) \sim \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

参数 μ 是平均值或中位数，决定了该分布的位置（正态分布是一种对称的分布）。参数 σ 决定了分布的幅度。这是一般意义上的方程式，在真实世界的各种特定现象中，这些参数的值是固定的，我们可以通过对数据的估计得到这些参数。

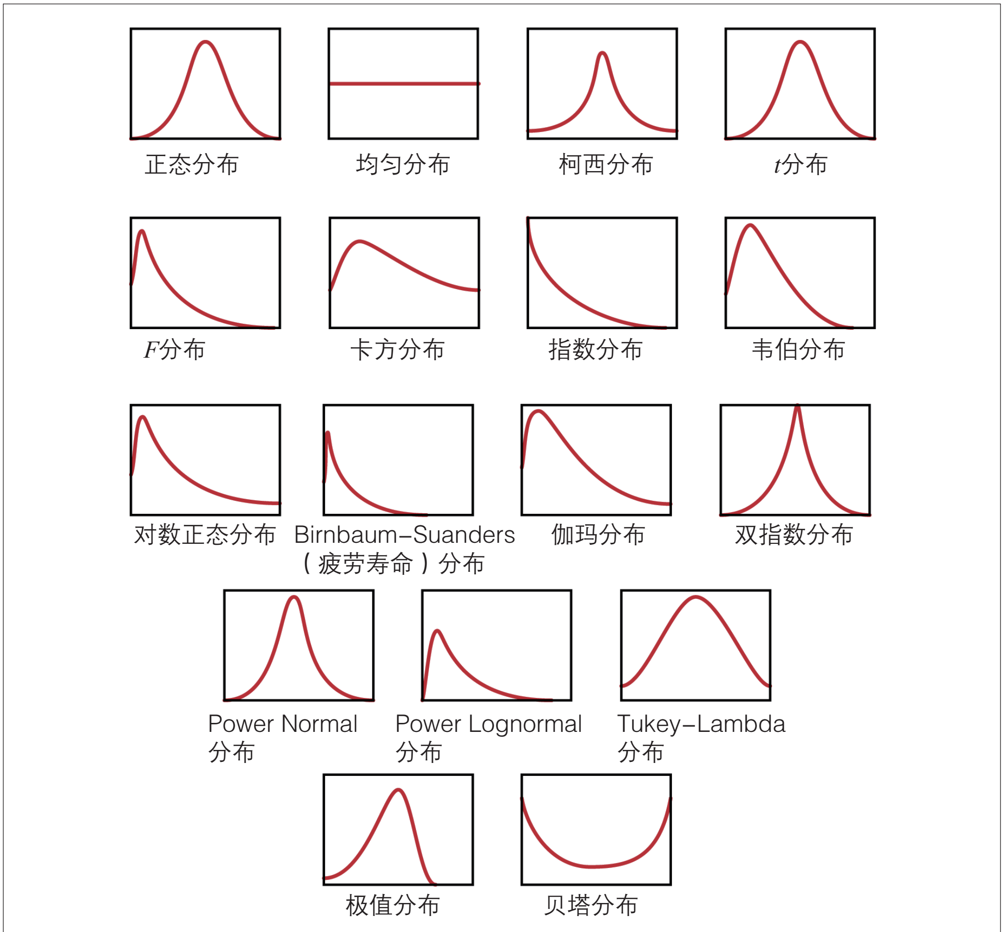


图 2-1：一组连续的密度函数（也叫概率分布）

假设随机变量 x 的概率分布为 $p(x)$，该函数将 x 映射为一个实数，要使其成为概率密度函数，需要对其做以下限定：使用积分求曲线覆盖下的面积，则其值必须为 1，这样才能称其为概率。

比如，设 x 为距离下趟公交车到站的时间（以秒为单位），则 x 是一个随机变量，因为下趟

公交车的到站时间是不定的。

假设我们已知（为了便于讨论）这个等待时间的概率密度函数为 $p(x)=2\mathrm{e}^{-2x}$，如果我们想知道下一趟车在等候 12~13 分钟后来的可能性，则只需要对于 12 至 13 之间该概率分布曲线下的区域使用积分 $\int_{12}^{13} 2\mathrm{e}^{-2x}$ 求面积即可。

怎么知道该使用哪种概率分布？有两种方法：首先，可以做实验。我们可以随机到达公交车站，测量等候下一趟公交车需要的时间，重复该实验多次。然后将测量得到的数据绘制成散点图，看看与哪种概率分布曲线吻合。或者基于我们对“等待时间”是一种普遍的自然现象这一事实的了解，马上会想到采用指数分布 $p(x)=\lambda\mathrm{e}^{-\lambda x}$ 去描述，指数分布就是专门发明用来描述自然界这种普遍现象的。

使用单变量函数可以描述一个随机变量的分布，描述多个随机变量则需要使用多变量函数，这称作联合分布。以两个随机变量为例，使用函数 $p(x, y)$ 表示概率分布，输入为平面上的点，输出为一个非负数。为了确保其是一个概率分布函数，在整个平面求二重积分，其值为 1。

还有一种分布叫条件分布 $p(x|y)$，其含义是当 y 给定时 x 的概率密度函数。

处理数据时，条件意味着一个子集。比如，假设我们有 Amazon.com 的一组用户数据，该数据列出了每个用户上月在该网站的消费金额，不论性别，也不管在将第一件商品加入购物车前浏览过多少商品。

设随机变量 X 表示消费金额，则可以用 $p(X)$ 表示用户的消费金额分布。

从所有用户中选取一个子集，该子集的用户在购买任何商品前至少浏览过 5 件商品，让我们看看这些用户上月消费金额的概率分布。设随机变量 Y 表示在购买第一件商品前浏览的商品数量，则 $p(X \mid Y>5)$ 表示条件分布。条件分布和一般的分布拥有一样的性质：求积分的结果为 1，而且永远不会为负数。

当我们观察这些数据点时，如 (x_1, y_1) ,(x_2, y_2),⋯, (x_n, y_n)，我们是在观察随机变量的实例。当我们有一个 n 行 k 列的数据时，我们是在观察 k 个随机变量组成的联合分布的 n 个实例。

如果读者还想了解更多关于概率分布的知识，请参考 Sheldon Ross 所著的 *A First Course in Probability* 一书（Pearson）。

拟合模型

拟合模型是指用观察数据估计模型参数的过程。以数据为依据，近似模拟现实中产生数据的数学过程。拟合模型经常要引入各种优化方法和算法，例如最大似然估计等，来确定参数。

事实上，当你估计参数时，参数就成了估计量，他们本身就是数据的函数。当模型拟合成功，就能以数学函数的形式表达，比如 $y = 7.2 + 4.5x$，其含义是，基于你对数据间存在线性关系的假设，该函数准确表达了两个变量之间的这种关系。

拟合模型的过程就是开始编写代码的过程：代码将会读入数据，将写在纸上的公式翻译成代码，然后使用 R 或者 Python 中内建的优化方法，根据数据，求出尽可能精确的参数值。

等你变得越来越老练，或者这本身就是你的强项时，你可能会去研究这些优化方法。首先得知道这些优化方法的存在，然后弄明白它们是怎么工作的，但是你不必亲自去编写代码实现这些方法，R 和 Python 已经帮你实现好了，直接调用就行。

过拟合

在本书中，不时会提醒你小心*过拟合*，甚至这会成为你的梦魇。过拟合是指使用数据去估计模型的参数时，得到的模型并不能模拟现实情况，在样本以外的数据上效果不好。

在试图用该模型去预测另一组数据（该组数据未用来拟合模型）的标签时，你可能会发现结果不尽如人意，以准确度去衡量，这并不是一个很好的模型。

2.2 探索性数据分析

> “面对那些我们坚信存在或不存在的事物时，“探索性数据分析”代表了一种态度，一种方法手段的灵活性，更代表了人们寻求真相的强烈愿望。”
>
> —— John Tukey

前面我们提到过探索性数据分析是建模的第一步。探索性数据分析经常是标准统计学入门教材的第 1 章（第 1 章的意思是，这是最简单、最初级的内容），然后全书就再也不会提及它，它被遗忘了。

探索性数据分析经常表现为画一些直方图或者茎叶图，小学五年级都开始教这些知识了，因此探索性数据分析看起来只是小菜一碟，不是吗？这也就难怪没人把它当回事了。

然而探索性数据分析是数据科学中的重要一环，同时代表了来自贝尔实验室的一批统计学家在从事数据科学工作时所采用的方法和观点。

John Tukey 是贝尔实验室的数学家，他开发出有别于验证性数据分析的探索性数据分析，如上节所述，验证性数据分析偏重于模型和假设。在探索性数据分析中，没有假设，也没有模型。这里的“探索性”是指你对待解问题的理解会随着研究的深入不断变化的。

回顾贝尔实验室的历史

贝尔实验室始于 20 世纪 20 年代，在物理学、计算机科学、统计学和数学上做出了很多重大发明和创新，程序设计语言 C++ 也诞生于贝尔实验室，该实验室还培养出很多诺贝尔奖获得者。这里有一支高产且成功的统计学小组，其中数学家 John Tukey 尤其著名，他解决了很多统计学问题。他被誉为探索性数据分析和 R 语言之父（R 的前身是贝尔实验室的 S 语言，R 是 S 语言的一个开源版本），他同时对高维数据的可视化也很感兴趣。

我们认为贝尔实验室是数据科学的诞生之地，因为那里有海量的复杂数据，还有各学科之间的相互合作。和现在的谷歌一样，那里过去是计算机科学家和统计学家的虚拟游乐场。

早在 2001 年，Bill Cleveland 在其文章"Data Science: An Action Plan for expanding the technical areas of the field of statistics"中就描述了多学科研究、模型、处理数据的方法（也就是传统的应用统计学）、和数据相关的计算（硬件、软件、算法、编码）、教学法、工具评价（现在依然是前沿科技）以及理论（数据背后的数学）。

读者可以阅读 Jon Gertner 所著的 *The Idea Factory*（Penguin Books）一书了解更多关于贝尔实验室的故事。

探索性数据分析的基本工具是图、表和汇总统计量。一般来说，探索性数据分析是一种系统性分析数据的方法，它展示了所有变量的分布情况（利用盒形图）、时间序列数据和变换变量，利用散点矩阵图展示了变量两两之间的关系，并且得到了所有的汇总统计量。换句话说，就是要计算均值、最小值、最大值、上下四分位数和确定异常值。

探索性数据分析不仅是一组工具，更是一种思维方式：要怎么看待和数据之间的关系。你想理解数据，了解数据的形状，获得对数据的直观感受，想将数据和你对产生数据的过程的理解关联起来。探索性数据分析是你和数据之间的桥梁，它不向任何人证明什么。

2.2.1　探索性数据分析的哲学

> "与其担心如何说服别人，不如先了解到底发生了什么。"
>
> —— Andrew Gelman

在谷歌期间，Rachel 有幸与前贝尔实验室的两位统计学家，Daryl Pregibon 和 Diane Lambert 共事，他们都是应用统计学领域的专家。正是从他们身上，Rachel 学会了将探索性数据分析作为她的最佳实践之一。

是的，即使面对谷歌级别的大体量的数据，他们依然进行探索性数据分析。在互联网企业中，基于和处理小数据同样的原因，探索性数据分析经常被用到，在处理日志数据时，有更多的理由使用探索性数据分析。

使用探索性数据分析有很多重要的原因。包括获取对数据的直觉、比较变量的分布、对数据进行检查（确保数据的规模在你预期范围内，数据的格式是你想要的等）、发现数据中的缺失值和异常值、对数据进行总结。

对于在日志中生成的数据，探索性数据分析可以用于调试记录日志的流程。比如，你通过统计日志数据发现的一些“模式”，很可能其实是由于日志记录流程中出错造成的，因此这些错误亟待修复。如果你怕麻烦从不去调试，你可能会一直认为这些模式是真实存在的。和我们工作过的工程师总是非常感谢我们在这方面提供的帮助。

最后，探索性数据分析确保了产品的性能符合预期。

在探索性数据分析中会引入许多图形，但是我们有必要在这里对探索性分析和数据可视化加以区分。探索性数据分析是数据分析的开端，而数据可视化（将会在第 9 章介绍）是在数据分析的最后一个环节，用于呈现数据分析的结论。在探索性数据分析中，图形只是帮助你理解数据。

在探索性数据分析中，可以根据对数据的理解优化算法。比如，你正在开发一种排名算法，该算法对你推荐给用户的内容进行排名。为此，你可能需要定义什么是“流行度”。

在决定以何种方式量化“流行度”之前（可行的量化方式有最高的点击率、最多的回复率、大于某一阈值的回复量或者众多指标的加权平均值），你需要先了解数据的运作表现，而做这件事最好方式就是观察你的数据，亲自去实践。

根据数据绘图，并进行比较，这些将会收到意想不到的效果。相比于拿到数据集后、不管三七二十一就运行一个回归模型，这种方法效果要好得多。你之所以选择回归模型，只是因为你知道它怎么用。对分析师和数据科学家来说，在处理数据时，若没有将探索性数据分析视为重要一环纳入到整个研究过程中，这对研究结果极为不利。给自己个机会，把探索性数据分析作为你的数据分析工作流程中的一部分吧！

这里有一些引用文献，帮助你了解探索性数据分析这一最佳实践和其历史背景：

(1) *Exploratory Data Analysis*，John Tukey 著（Pearson）

(2) *The Visual Display of Quantitative Information*，Edward Tufte 著（Graphics Press）

(3) *The Elements of Graphing Data*，William S. Cleveland 著（Hobart Press）

(4) *Statistical Graphics for Visualizing Multivariate Data*，William G. Jacoby 著（Sage）

(5)“Exploratory Data Analysis for Complex Models”，Andrew Gelman（American Statistical Association）

(6) John, T. (1962). The Future of Data Analysis. *Annals of Mathematical Statistics*, 33(1), 1-67

(7)“Data Analysis, Exploratory”，作者为 David Brillinger，*International Encyclopedia of Political Science (Sage)*（8 页摘录）

2.2.2 练习：探索性数据分析

有 如 下 31 个 数 据 文 件：nyt1.csv、nyt2.csv…nyt31.csv， 可 以 从 https://github.com/oreillymedia/doing_data_science 下载。每一个数据文件记录了《纽约时报》5 月份每天出现在主页上的广告和广告的点击次数，当然，这组数据是我们伪造的。数据的每一行代表了一个用户，数据共有 5 列，分别为：年龄、性别（0 = 女性，1 = 男性）、广告显示次数、点击次数、是否登录。

你将使用 R 处理这些数据，R 是一种编程语言，设计用来专门做数据分析，使用起来很直观。如果你还没安装过 R，请致其官方网站 http://www.r-project.org/ 下载。安装完成后，使用下述命令加载一个文件：

```
data1 <- read.csv(url("http://stat.columbia.edu/~rachel/datasets/nyt1.csv"))
```

数据加载成功后，就可以开始进行探索性数据分析了。

(1) 创建一个新变量 age_group ，按年龄将用户离散化，分为 `"<18"`、`"18-24"`、`"25-34"`、`"35-44"`、`"45-54"`、`"55-64"`、`"65+"` 总共 7 组。

(2) 对于每天的记录有以上操作。

- 对这 7 组用户，分别绘出点击率分布图 (点击率 = 点击次数 / 广告显示次数)。
- 定义一个新变量，基于用户的点击行为将用户分类。
- 探索性地分析这些数据，从图形和数量两方面比较各用户组之间的差异（比如小于 18 岁的男性和小于 18 岁的女性，或者已登录用户和未登录用户等）。
- 创建用于描述数据的各种统计量、矩阵。可能的度量项目有点击率、分位数、平均值、中位数、方差、最大值等，可以在每个用户组中单独计算这些统计量。要有所取舍，想一想随着时间的推移，哪些才是重要的、值得去记录的，这样可以压缩数据，同时不失准确地记录用户行为。

(3) 现在延长你所分析的时间，将数日内的矩阵和分布情况用图形表示出来。

(4) 描述并解释你发现的模式。

示例代码

这里我们给出本练习的一个解决方案的前半部分代码。我们不可能在这一本书中同时教授数据科学和如何编写程序，学习使用一种新语言去写程序需要不断去尝试，你可能会遇到错误，这时去谷歌或 stackoverflow 网站上搜索是个不错的主意。

当你试图在 R 中绘图或者建模是，或许其他人已经试过了，与其一个人在那苦思冥想，不如上网看看[1]。但是在你努力凭自己的能力解决这个问题之前，我们不建议马上看我们提供的答案：

注 1：O’Reilly 网站上也有很多参考书可供阅读。

```
# Author: Maura Fitzgerald
data1 <- read.csv(url("http://stat.columbia.edu/~rachel/datasets/nyt1.csv"))

# 分类
head(data1)
data1$agecat <-cut(data1$Age,c(-Inf,0,18,24,34,44,54,64,Inf))

# 预览
summary(data1)

# 分组
install.packages("doBy")
library("doBy")
siterange <- function(x){c(length(x), min(x), mean(x), max(x))}
summaryBy(Age~agecat, data =data1, FUN=siterange)

# 登录的用户才有性别和年龄
summaryBy(Gender+Signed_In+Impressions+Clicks~agecat,
          data =data1)

#绘图
install.packages("ggplot2")
library(ggplot2)
ggplot(data1, aes(x=Impressions, fill=agecat))
      +geom_histogram(binwidth=1)
ggplot(data1, aes(x=agecat, y=Impressions, fill=agecat))
      +geom_boxplot()

# 根据转化率创建点击，如果没有印象，
# 不必在乎其是否点击，如果没有印象的点击出现，
# 证明我对数据的假设是错误的
data1$hasimps <-cut(data1$Impressions,c(-Inf,0,Inf))
summaryBy(Clicks~hasimps, data =data1, FUN=siterange)
ggplot(subset(data1, Impressions>0), aes(x=Clicks/Impressions,
       colour=agecat)) + geom_density()
ggplot(subset(data1, Clicks>0), aes(x=Clicks/Impressions,
       colour=agecat)) + geom_density()
ggplot(subset(data1, Clicks>0), aes(x=agecat, y=Clicks,
      fill=agecat)) + geom_boxplot()
ggplot(subset(data1, Clicks>0), aes(x=Clicks, colour=agecat))
      + geom_density()

# 分组
data1$scode[data1$Impressions==0] <- "NoImps"
data1$scode[data1$Impressions >0] <- "Imps"
data1$scode[data1$Clicks >0] <- "Clicks"

# 将列转换成一个因素
data1$scode <- factor(data1$scode)
head(data1)

#查看水平
clen <- function(x){c(length(x))}
etable<-summaryBy(Impressions~scode+Gender+agecat,
                  data = data1, FUN=clen)
```

以下为做该练习其他部分时的提示：

> 不要将所有数据一次性读入内存。当你某天终于将代码写好时，一次只加载一个数据文件，处理它，输出相关的矩阵和变量，将结果存入一个数据框。在加载下一个文件时，记得移除上一个文件。之所以这样做，是为了让你思考在多个机器之间共享数据时该如何处理。

关于编写代码的建议

在 2013 年 5 月发表的一篇文章“How to be a Woman Programmer”中，Ellen Ullman 相当好地描述了成为一个程序员需要的素质（让我们暂且不去关注和女性特别相关的部分）：

学习编程的首要条件是对编程本身的热爱，对于探索横亘于人脑和机器之间的神秘空间、如何使机器满足人类的需求抱有强烈的探索需求。

第二个条件是允许失败。编程是一门设计算法的艺术，同时是一门调试错误程序的手艺。用 Fortran 语言的发明者、著名的计算机科学家约翰·巴克斯的话来说：“你要随时准备着出错，你要有很多方案，努力工作去发现那些不奏效的方案，不断地这样做，直到找到正确的方案。”

2.3　数据科学的工作流程

综上所述，让我们来看看如何定义数据科学的工作流程。你见的数据科学工作者越多，你就越会发现他们的工作流程符合图 2-2 的描述。贯穿全书，我们会使用各种方法介绍该过程的各个环节和案例。

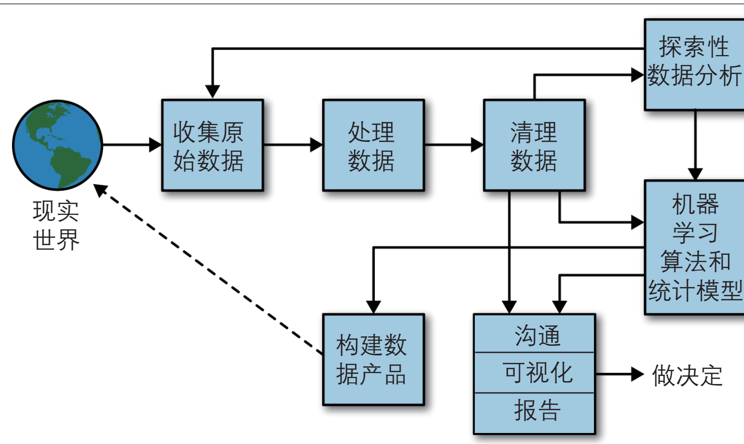


图 2-2：数据科学的工作流程

首先，我们生活在这个世界中。在这个世界上，有很多人在从事各种各样的活动。有些人在使用 Google+，另外一些人则在奥运会上一较高下；有些人在制造、发送垃圾邮件，有些人则在医院里抽血。假设我们拥有其中某项活动的数据。

具体来说，以原始数据为起点，诸如日志、奥运会纪录、安然公司员工的电子邮件、遗传物质记录（需要注意的是，在我们拿到这些原始数据时，这项活动中某些方面的信息已经缺失了）。我们需要处理这些原始数据，使得其便于分析。因此我们创建出管道对数据进行再加工：联合、拼凑、清理，随便你叫它们什么好了，就是要对数据进行再加工。我们可以使用 Python、shell 脚本、R、SQL 完成这件任务。

最终得到格式化好的数据，像下面这种由列构成的数据：

姓名 | 事件 | 年份 | 性别 | 时间

> 在标准的统计学课程中，通常从一份干净有序的数据文件开始，但在现实中，你通常不会有这么好的运气。

在拿到这份干净的数据后，我们应该先做一些探索性数据分析。在这个过程中，我们或许会发现数据并不是那么干净，数据可能含有重复值、缺失值或者荒谬的异常值，有些数据未被记录或被错误地记录。在发现上述现象时，我们不得不回过头采集更多的数据，或者花更多的时间清理数据。

然后，我们使用一些算法，比如 *k* 近邻、线性回归、朴素贝叶斯等设计模型。选取何种模型取决于要解决的问题，这可能是一个分类问题、一个预测问题，或者只是一个基本的描述问题。

这时就可以解释、勾勒、报告或者交流得到的结果。可以将结果报告给老板或同事，或者在学术期刊上发表文章，或者走出去参加一些学术会议，阐述我们的研究成果。

如果我们的目标是开发一款数据产品或其产品原型，例如垃圾邮件分类、搜索排名算法、推荐引擎等。数据科学和统计学的不同之处就体现出来了，数据产品最终会融合到日常生活中，用户会和产品产生交互，交互会产生更多的数据，这样形成一个反馈的循环。

这和天气预报大相径庭，在预测天气时，你的模型对于结果没有任何影响。比如，你预测到下星期会下雨，除非你拥有某种超能力，否则不是你让天下雨的。但是假如你搭建了一个推荐系统，证明“很多人都喜欢这本书”，那就不一样了，看到这个推荐的人没准觉得大家都喜欢的东西应该不会太差，也喜欢上这本书了，这就形成了反馈。

在做任何分析时，都要将这种反馈考虑在内，以此对模型产生的偏差进行调整。模型不仅预测未来，它还在影响未来。

一个可供用户交互的数据产品和天气预报分别处于数据分析的两个极端，无论你面对何种类型的数据和基于该数据的数据产品，不管是基于统计模型的公共政策、医疗保险还是被广泛报道的大选调查，报道本身或许会左右观众的选票，你都要将模型对你所观察和试图理解的现象的影响考虑在内。

数据科学家在数据科学工作流程中的角色

到目前为止，所有这一切仿佛不需要人工干预，奇迹般地发生了。这里说的“人”，是指那些“数据科学家”。总得有人做出决定：该收集哪些数据？为什么要收集这些数据？她还要提出问题，做出假设，制定解决问题的方案。她就是数据科学家，或者她是我们推崇的数据科学团队。

让我们重新修订以前的流程，至少增加一层，来表明数据科学家需要全程参与到这一流程中来，他们不但需要在流程的较高层次上工作，还需要亲手编写程序，如图 2-3 所示。

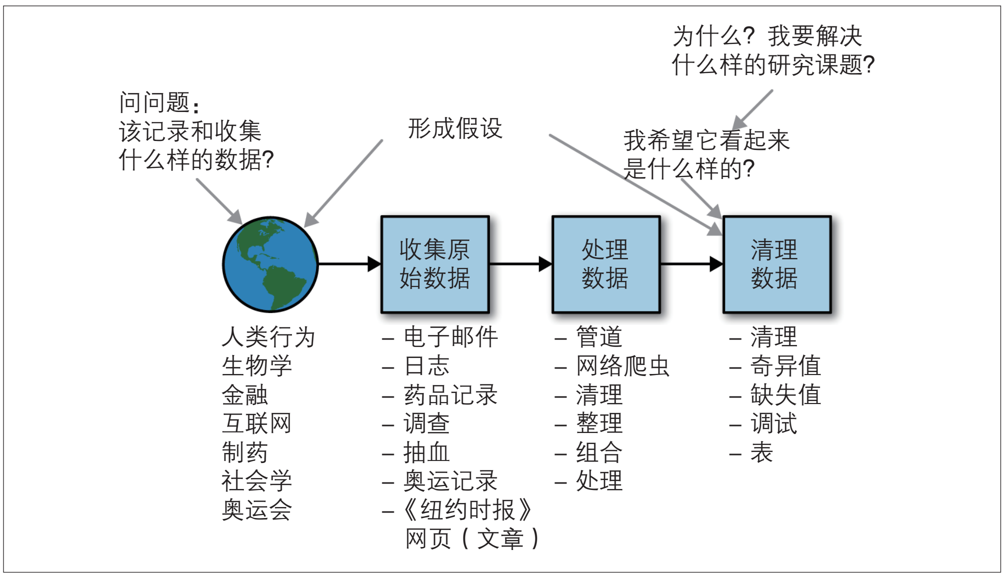


图 2-3：数据科学家需要参与数据科学工作流程的各个环节

数据科学工作流程和其他科学方法的关系

数据科学工作流程，可以看作是其他科学方法的延伸或变体，它的一般步骤为：

- 提出问题；
- 做一些背景研究；
- 构想假设；

- 做实验验证构想的假设；
- 分析数据并得出结论；
- 把你的结果分享给其他人。

在数据科学工作流程和其他科学方法中，不是每个研究问题都需要按部就班地解决，大多数问题都不用严格走完每一步，几个步骤的组合就可能解决问题。比如，如果你的目标是对数据进行可视化（这本身也可以看成是一个数据产品），很可能你不会使用任何机器学习或统计模型，你只需要想方设法得到干净的数据，做一些探索性数据分析，将结果用图表的形式展示出来即可。

2.4 思维实验：如何模拟混沌

大多数问题一开始都面临一堆脏乱无序的数据，或者问题本身并未得到明确定义，或者问题迫切待解。作为数据科学家的我们，从某种程度上说，肩负着从混沌中恢复秩序的责任。在哥伦比亚大学的课堂上，我们利用课间休息时间讨论了如何来模拟混沌，下面是讨论中出现的一些有意思的观点。

- 洛伦兹水车是一种摩天轮式的精妙装置，它由等间距的叶轮组成，绕轴旋转。每个叶轮下方都有一个小孔，设想一下水流从水车的正上方倾泻而下，从小孔中漏出的水会打到其他叶片上。调整水的流速，会发现叶轮一会儿正转，一会儿反转，呈现一种混沌的状态，请阅读维基百科上的文章了解更多关于洛仑兹水车的内容：http://en.wikipedia.org/wiki/User:Pankajgarg_india/The_lorenzian_waterwheel。
- 很多系统可以演示固有的混沌。Philippe M. Binder 和 Roderick V. Jensen 写过一篇关于利用计算机模拟混沌现象的文章，文章的题目是“Simulating chaotic behavior with finite-state machines”。
- 来自麻省理工、哈佛和塔夫茨大学的研究者发起了一个跨学科的项目，旨在教授“Simulating chaos to teach order”这项技术。他们模拟了发生在乍得和苏丹边境的达尔富尔地区的一起紧急事件，学生在其中扮演无国界医生组织成员、国际医疗队和其他一些人道主义机构。
- Joel Gascoigne 也写了一篇关于混沌的文章“Creating order from chaos in a startup”。

讲师笔记

(1) 在一个组织中充当数据科学家经常要面对各种混乱，从混乱中恢复秩序是数据科学家的职责。因此，在课堂上我会不断地给我的学生模拟各种混乱的场景。但愿学生们明白这只是出于教学的目的，并非老师无能。

(2) 我想通过对“混沌”这个词的不同理解，来阐述词汇的重要性和与人沟通时的困难，人们经常并不知道一个词的确切含义，或者他理解的和你想说的南辕北辙。数据科学家要和领域专家沟通，他们很可能不知道什么是“逻辑回归”，但为了不让自己看起来像个傻瓜，或者觉得这是他们“应该”知道的，总之，他们装作了然于胸，不屑向你提问。若两个人在讨论时并不确知对方口中的术语是什么意思，这样的沟通怎么可能有效？同样的，数据科学家也应该多问领域专家问题，确保自己理解领域专家用到的术语（不管他是一位天体物理学家、社交网络专家还是气象学家）。不知道某些术语并不丢人，不知道还不去问才丢人。你很可能会发现，当通过提问题搞清楚一些术语的含义后，你会对问题本身理解得更透彻。

(3) 模拟是数据科学中一项极为有用的技术。为一个模型模拟一些假数据可以更好地理解产生数据的过程，还可以帮助调试程序，这是一项很好的实践。

2.5 案例学习：RealDirect

RealDirect 公司的 CEO Doug Perlson 熟习房地产法，有初创公司和在线广告领域的工作背景。他创立 RealDirect 公司的目标是利用收集到的房地产数据帮助人们买卖房子。

通常人们每七年就会卖掉自己的房子，这经常要借助于房产经纪人和当前的市场数据。但是房产经纪系统和数据质量本身都存在不少问题，RealDirect 正是致力于解决这两方面的问题。

首先说房产经纪人，他们经常是“自由代理人”，他们给自己打工，你可以把他们想象成房产销售顾问。这就意味着他们会极力保护自己手中的数据，那些业绩很好的经纪人通常手中握有大量的数据。但是长远来看，这只不过意味着他们比那些菜鸟多掌握一些数据而已。

为了解决这些问题，RealDirect 雇用了一些持有执照的房产经纪人，他们一起工作，共享各自掌握的信息。RealDirect 给卖房者提供了接口，给他们一些基于统计数据的卖房建议。RealDirect 还利用和用户的交互数据，实时地指导用户该如何进行下一步操作。

这些被雇用的房产经纪人现在成了数据专家，他们学着利用一些数据采集工具收集有用的新信息，或者直接访问那些公众可见的数据。比如，现在你能获取合作式公寓（纽约的一种公寓）的销售数据，这得益于最近的政策变化。

公开的数据有一个缺点：时效性不强。一笔房屋交易完成后，通常要等三个月左右才会被录入公开可查询的数据库中。而 RealDirect 正致力于向用户提供实时反馈，包括用户何时开始搜索房屋、初始报价是多少、从放盘到成交需要多长时间、用户是如何在网上搜索房屋的……这一系列问题，RealDirect 都能为用户提供有效信息。

最终，如果买方和卖方都诚实守信，这些优质的信息对双方都有帮助。

2.5.1 RealDirect是如何赚钱的

首先，它向卖方提供每月大概要花费 395 美元的订阅服务，用以访问网站提供的销售工具。其次，它允许卖方以较低的价格使用公司的房产经纪人，通常是房屋总价的 2%，而市面上的价格一般是 2.5% 或 3%。这时，共享信息的优势就体现出来了，这种更优化的人力资源组织方式允许 RealDirect 采取更低的定价策略，从而带来更多的生意，利润自然增加了。

RealDirect 网站更像一个平台，在这个平台上，买方和卖方可以管理他们的买卖流程。网站实时反映了用户的当前状态：活动、交易达成、交易被拒绝、看房中、正在签合同中等。软件会根据用户的当前状态给出下一步行动的建议。

当然，RealDirect 也面临一些挑战。首先，根据纽约的法律，只有登记造册的房屋才能出现在出售目录上，因此 RealDirect 网站需要用户注册。对于买家来说，这无疑是人为地被设置了一道障碍，但是那些真正想买房子的人，是不会因为这点小麻烦而放弃注册的。此外，那些不需要注册的网站，比如 Zillow，并不是 RealDirect 的竞争对手，因为它们只提供出售的房屋目录，并不提供其他附加服务。Doug 指出，使用 Pinterest 同样需要注册，但依然有无数用户去注册，因此，需要用户注册对 RealDirect 来说并不是什么问题。

RealDirect 的房产经纪人来自各个房产经纪机构，即使这样，RealDirect 仍然收到一些来自房产经纪人的来信，他们谴责 RealDirect 单方面降低价格的行为损害了整个行业的利益。但同时，如果一个房产经纪人因为某房产在 RealDirect 网站上售卖，而拒绝带领买主去看房的话，这些潜在的买家就会抱怨，他们也在其他地方看见了该房产，凭什么不让看房。因此，传统的房产经纪人别无选择，即使他们不喜欢 RealDirect，也不得不和它做生意。换句话说，这些房屋销售的目录是透明的，传统的房产经纪人没有办法不让他们的买家看见这些房子。

Doug 谈到了买家购房时考虑的一些关键因素：附近有没有公园、地铁、学校，同片区或同建筑内相似公寓每平米价格的差异。他们想收集这些数据用到 RealDirect 的服务中。

2.5.2 练一练：RealDirect公司的数据策略

你被任命为 RealDirect 公司的首席数据科学家，直接汇报给 CEO。公司（当然是假想的公司）现在还没有关于如何利用数据的计划，全指望你提出一套数据策略。下面是一些能帮助你开始指定策略的一些方法。

(1) 浏览 RealDirect 网站，站在用户的角度，想一想买方和卖方分别会如何浏览在各页面、网站的各个页面之间是如何组织的。试着去理解现有的业务模型，想一想如何通过分析 RealDirect 网站用户行为，帮助企业做出决策、改善产品。提出一些你认为可以用数据回答的问题。

- 你会建议工程师为哪些数据记录日志？你期望的数据文件应该拥有何种格式？
- 如何使用数据汇报和管理产品的使用？
- 从数据中得到的信息又如何反馈给产品和网站？

(2) 因为现在还没有数据供你分析（通常在初创型公司中，产品这时还处于开发阶段），此时，你应该借助于一些辅助数据来获取对市场的认识。比如，可以去如下网站下载一些数据文件：https://github.com/oreillymedia/doing_data_science。
你可以使用部分或全部的数据文件。

- 第一个挑战：加载和清理数据。然后进行探索性数据分析，找出哪些数据中有异常值和缺失值，你会采取何种方式处理它们？最后，确保将数据格式转换为你想要的格式，诸如你认为整数类型的值应该确保其类是整型等。
- 当数据整理干净后，继续探索性分析，将数据间的关系用图形展示出来，将数据在 (i) 空间和 (ii) 时间两个维度上进行对比。如果你时间充裕，可以试着找找这些数据里面是否蕴含着什么有意思的模式。

(3) 将你的发现总结成一份简报汇报给 CEO。
(4) 作为数据科学家工作的一部分，经常需要向那些不是数据科学家的人发表讲演，因此理想情况下，你需要掌握一些沟通的技巧，可以将你要表达的信息准确传达给对方。你能想到还应该和哪些人进行交流吗？
(5) 大多数人不是房地产行业或电子商务的“领域专家”。
    - 跨出自己的舒适区，学会在一个不同的环境中采集数据是否给了你一些启示？使得你知道如何在自己的领域内行事。
    - 有时候“领域专家”有他们专有的词汇。Doug 是否使用了一些他领域内的专有词汇是你所不理解的（“comps”“open houses”“CPC”）？有时候，如果你不明白一些专家正在使用的词汇，会妨碍你搞清楚问题。养成提问的好习惯，因为迟早你会碰到一些你不明白的事，这需要持之以恒。

(6) Doug 提到他的公司并不是必须要有一个数据策略。也没有指定数据策略的业界标准。在你做这个练习的过程中，考虑一下是否有一些你想推荐的最佳实践，可以为电子商务或者你自己的领域指定数据策略提供帮助。

示例R代码

下述 R 代码以上节用到的布鲁克林区的房屋销售数据为例，进行了数据清理和探索性数据分析（练习要求使用曼哈顿的数据）。

```
# 作者：Benjamin Reddy

require(gdata)
bk <- read.xls("rollingsales_brooklyn.xls",pattern="BOROUGH")
head(bk)
```

```
summary(bk)

bk$SALE.PRICE.N <- as.numeric(gsub("[^[:digit:]]","",
                              bk$SALE.PRICE))
count(is.na(bk$SALE.PRICE.N))

names(bk) <- tolower(names(bk))

## 使用正则表达式清理和格式化数据
bk$gross.sqft <- as.numeric(gsub("[^[:digit:]]","",
                            bk$gross.square.feet))
bk$land.sqft <- as.numeric(gsub("[^[:digit:]]","",
                           bk$land.square.feet))

bk$sale.date <- as.Date(bk$sale.date)
bk$year.built <- as.numeric(as.character(bk$year.built))

## 做一些探索性分析
## 确保销售价格无异常出现
attach(bk)

hist(sale.price.n)
hist(sale.price.n[sale.price.n>0])
hist(gross.sqft[sale.price.n==0])

detach(bk)

## 只保留有价值的订单
bk.sale <- bk[bk$sale.price.n!=0,]

plot(bk.sale$gross.sqft,bk.sale$sale.price.n)
plot(log(bk.sale$gross.sqft),log(bk.sale$sale.price.n))

## 先看 1、2 和 3 家庭住宅
bk.homes <- bk.sale[which(grepl("FAMILY",
            bk.sale$building.class.category)),]
plot(log(bk.homes$gross.sqft),log(bk.homes$sale.price.n))

bk.homes[which(bk.homes$sale.price.n<100000),]
    [order(bk.homes[which(bk.homes$sale.price.n<100000),]
    $sale.price.n),]

## 去除那些看起来不像真实订单的奇异值
bk.homes$outliers <- (log(bk.homes$sale.price.n) <=5) + 0
bk.homes <- bk.homes[which(bk.homes$outliers==0),]

plot(log(bk.homes$gross.sqft),log(bk.homes$sale.price.n))
```

第 3 章

算法

上一章探讨了数据科学中模型应用的概况，本章我们将深入学习算法。

算法是完成分析任务所采纳或者遵循的一整套步骤和规则，它是计算机科学中一个基本概念，可视作计算机科学的基石。设计优雅高效的代码、准备和处理数据以至软件工程开发均以算法为基础。

排序、查找、基于图的计算等问题都是算法能够解决的。然而，对于同一个问题，基于效率和计算时间的考虑，可以选出某个相对最优的算法。当算法要解决的数据分析问题涉及海量数据，或者开发面向客户的产品时，基于效率选择最优的算法会变得尤为重要。

高效的算法是准备和处理数据的基础，算法的执行可以是顺序的，也可以是并发的。就数据科学来说，以下三类算法是必须了解的。

(1) 数据清理和预处理的算法。比如排序、MapReduce、Pregel。

  我们将这类算法称为数据工程，我们将用一章内容专门介绍这类算法，然而，这不是本书的重点。这并不代表你永远不会用到它们，只是作为一类算法，本书并不予以特殊强调。

(2) 用于参数估计的最优化算法。比如 Stochastic Gradient Descent（随机梯度下降）、Newton’s Method（牛顿法）和 Least Squares（最小二乘法）。在本书中，我们会介绍这些算法。另外值得一提的是，在 R 软件中这些算法都有相应的实现函数。

(3) 机器学习算法。该类算法是本书的重点，也将占据本书的大部分篇幅，下面我们就详细介绍该类算法。

3.1 机器学习算法

机器学习算法的应用主要有三个大舞台：预测、分类和聚类。

且慢！在上一章中你不是刚刚说过，是用模型来做这些事情吗？确实！这里似乎有一点混乱，我们先来澄清一些概念。

统计*模型*出自统计学家之手，机器学习*算法*则是计算机科学家所开发的。但某些技术和方法在统计模型和机器学习算法之中都会用到。因此，在本书中这两个词有时可以换着使用。

你会发现本书中的一些算法，比如线性回归，在机器学习和统计学的书中都会出现。讨论这些算法到底来源于哪个学科没有太大的实际意义。

一般来讲，机器学习是人工智能的基础，比如图像识别、语音识别、推荐系统、排名和个性化推荐系统技术——这些都是打造数据产品的基础。然而传统的统计学院里可能不会设置这些课程。因为这些算法的作用往往不是为了推断数据背后的生成过程（这是统计推断的目的所在），而是为了尽可能准确地预测和分类。

Rachel 在谷歌工作期间，以及参加一些学术会议时，发现机器学习专家和统计学家在处理问题的方式上也有所不同，这种不同反应了各自学科文化上的差异。但作为数据科学家，应该能够在两种思维模式下自由的切换。

这里我们澄清一些基本概念。

- *关于参数的解释*

  统计学家认为模型中的参数必须在现实世界中是有意义的。就拿线性回归模型来说，模型中的参数往往都对应着现实世界中的某种行为或者现象，因此模型可以看做是现实世界的某个缩影。而参数在软件工程师或计算机科学家的眼中又是另一番景象，他们主要关心的是如何将算法（包含其中的参数）植入到数据产品中，模型有些可以很复杂，甚至难以解释。有些模型甚至被叫作黑匣子，因为他们根本不知道模型内部到底是如何运作的。他们通常不会关注模型参数的意义。即便他们想要关注，也可能是只是为了提升模型的预测能力，而不是为了解释这些参数。

- *置信区间*

  在统计学中，置信区间和后验分布用来描述参数估计的不确定性。但是，有些机器学习算法，比如 k 均值算法（k-means）和 k 近邻算法（k-nearest neighbors），就不涉及置信区间和参数估计的不确定性问题。我们稍后会详细介绍这两个算法。

- *显式假设的角色*

  统计模型会对数据的生成过程和数据的分布做出一些明确的假设（称作显式假设），统

计推断的过程往往是建立在这些假设的基础之上。而本章稍后将要介绍的非参数检验方法，并不对概率分布做任何显式假设（可能是隐式的）。[1]

可以说，统计学家每天都在和“不确定性”打交道，对任何事情他们都不会 100% 的确信。而软件工程师喜欢打造产品，他们喜欢构建模型以尽可能精确地做出预测，但他们可能并不关心模型本身的不确定性——只要模型效果好就行！像 Facebook 和谷歌这样的公司，其理念就是打造产品，并随着新数据的涌入不断迭代与更新产品。一个数据科学家要能够在统计学和计算机科学的思维方式之间找到一个平衡点，要取长补短。数据科学家的身上同时流淌着统计学家和计算机科学家的血液，大可不必厚此薄彼。最后让我们引用客座讲师 Josh Wills 的话对本节做一总结，他的这番话被推特上很多人引用过：

> “数据科学家是软件工程师中最好的统计学家，是统计学家中最好的软件工程师。”
>
> —— Josh Wills

3.2 三大基本算法

对于数学科学家来说，商业和现实世界中的很多问题都可以转化为相应的数学模型的形式，最终都可以归类为分类和预测两大问题。学术界和工业界对这两大问题的研究已经比较成熟了，有很多的算法可以直接拿来用。

作为一个数据科学家，在熟练掌握各种算法之后，真正的挑战其实才刚刚开始。你需要根据特定问题和隐含的假设来敲板到底使用哪个算法或者模型。这种判断部分源自经验：当解决的问题足够多时，每当遇到一个新的问题你可能会思考：“这是一个典型的分类问题，模型的输出变量是一个二元变量”，“这应该也是一个分类问题，但奇怪的是输出变量没有做任何标签”。同样都是分类问题，你却知道应该使用不同的算法处理。（第一个问题，你可能会选择逻辑回归或者朴素贝叶斯模型，第二个问题可以采用 k 均值算法。稍后我们会更详细的介绍这些算法。）

当你还是个学生或初学者时，乍听这些算法可能会有点不知所措，你时常会想“我怎么知道我想解决的这个问题需要使用哪种算法”。这些都是建模的内功，是需要花时间和精力去领悟的。

有一种人拿着锤子，觉得满世界都是钉子。“我会线性回归，碰到的所有问题我都想用线性回归模型去解决。”千万不要这样，作为一个合格的数据科学家，要不断尝试去了解问题的上下文和问题本身的属性，把它们用数学模型的形式表达出来，然后再想想你学习过的算法中哪些可以用来解决该问题，哪些更加适合这个问题。

注 1：比如说，分布是对称的，或者光滑的。

如果你还是心里没底，找个懂行的人去深入探讨一番也无妨。问问你的同事，参加一个讨论组，或者在你的周围发起一个这样的讨论组。问题之所以成为问题，就在于它的解决方案不是显而易见的。要时刻保持这样虚心的态度，这样当你去解决一个问题时，就会更加审慎。你不必成为一个建模领域的"百事通"，不要说："显然，这个问题使用带有惩罚项的线性回归模型就能解决。"千万别轻易下结论，即使有时你认为答案很明显，也要多听听旁人的意见。

之所以说起这些，是因为我们总是笃信教科书教给我们的。教科书总是把问题和解决这些问题的方法都摆出来，然后告诉你哪类问题应该用哪种方法。（比如，用体重预测身高应该使用线性回归模型。）这样的教科书学习模式对于刚开始理解和学习线性回归模型是有一定好处的：因为新的知识被吸收是需要一定时间的，需要多加练习。但是，当你熟练掌握了这项技术后，真正的挑战在于你能否从一开始就要知道什么情况下应该使用线性回归模型。这里面存在的挑战往往是教科书不会告诉我们的。

本书不会讨论所有的机器学习算法，毕竟本书不是一本有关机器学习的书。关于算法的书市面上已经有很多了，而且有些写得非常好。

话虽如此，我们还是会在本章介绍三种基本的算法，随着本书后续内容的展开，还会介绍其他算法。学习这些算法的目的其实在于培养举一反三的能力：在遇到新的、非教科书式的问题时，我们要起码能看清解决问题的可能方向。

我们会在书中尽量展示数据科学家在面对一个问题时的思考过程，结合问题的上下文，该如何确定该使用哪种算法。我们建议读者在遇到一个数据问题时，习惯性的让自己思考：这个问题的属性是什么？这些属性如何影响到算法的选择？

这里先介绍一些基本的机器学习算法，让我们以线性回归、k 近邻（k-NN）和 k 均值作为开始。就像之前说的，算法的选择与问题的属性相关，要看这些算法是否适合解决该问题。同时也应该从另一个角度审视这些算法，哪些模式是我们仅凭肉眼就可以发现的？当数据变得复杂时，光凭眼睛观察数据中的模式会变得不现实，这时候，就要及时借助计算机的帮助了。

3.2.1　线性回归模型

线性回归是统计学中最常用的算法之一。从根本上来说，当你想表示两个变量间的数学关系时，就可以使用线性回归。当你使用它时，你首先假设输出变量（有时称为响应变量、因变量或标签）和预测变量（有时称为自变量、解释变量或特征）之间存在线性关系。当然这种线性关系也可能存在于一个输出变量和数个预测变量之间[2]）。

注 2：这称作多元线性回归。

到底是算法还是模型？

在本章的开始，我们就两者的差别做过说明。从定义上来说，两者是完全不同的，然而在日常使用时却常常可以交替使用，这给大家带来了些许困扰。严格来说，算法是完成某项任务时需要遵循的一组规则或步骤，而模型是对世界一种附有假设的描述。这两个概念看起来显然是不同的，它们的区别也应该是显而易见的。然而，现实来看并非如此。比如，回归可以是一个统计学模型，也可以是一种机器学习算法。我们觉得，要精确区分两者之间差别，是在浪费时间，毫无必要。

从某种程度上说，这是个历史遗留问题。统计学和计算机科学一直在并行发展，他们常常使用不同的词汇描述同样的东西。这也就导致很难确定某个概念到底是机器学习算法还是统计模型。有一些方法（比如下一节要讨论的 k 均值）我们称为算法，从统计学的角度来看，它又是一种特殊的高斯混合模型。

因此我们建议，当人们谈起这些方法时，既可以说事算法也可以说是模型，尽量不要让这些干扰到你。（其实在这一行混这么久了，我们也时常对此感到困惑。）

输出变量和预测变量之间存在线性关系是一个大胆的假设，同时也是一个最简单的假设。从数学表示形式来看，线性函数比非线性函数更加基本。在解决问题的时候，我们总是从最简单的模型开始着手，线性模型是个不错的选择。

即便简单，但是线性假设有时候也不无道理。有时候，一个变量的变化和另一个变量的变化的确是线性的。比如，你卖的伞越多，你赚的钱越多。此时，你可以充分相信自己做出的线性假设是合理的。而有时候，确认变量之间的线性关系是困难的。但是微积分的角度来说，只要函数是连续的，函数可以被一些局部线性的函数所拟合。因此，局部线性假设大多数情况下都是合理的，但是全局线性假设就很难说了。

线性模型可能适用于类似下面的一些问题：比如说你正在研究一个公司的销售额和该公司在广告上的投入之间的关系，或者某人在社交网站上的好友数量和他每天在该社交网站上花费的时间之间的关系。这些问题的输出变量都是数值型的，也就意味着线性回归模型可能是一个不错的选择。最起码可以说，我们从线性模型作为研究的起步是没有太大问题的。

理解线性回归的一个切入点是先来确定那条直线。我们知道，通过斜率和截距就可以完全确定一条直线 $y = f(x) = \beta_0 + \beta_1 x$，但是，这样的一条直线是完全确定的。[3]

考虑随机函数，即便是对于数学功底不错的我们来说，也是一个新鲜的概念，但是在数据科学中却再常见不过了。然而，随机函数从根本上来说，还是建立在确定型函数的基础之上的。因此，我们不妨先讨论一下确定型函数的概念，用几个例子来切身感受一下。

注 3：这样的确定型函数（模型）对于数据分析来说起到了方向性的作用，但是数据往往是有很大噪声的，这就需要在确定型函数的基础之上考虑噪声，也就是随机性的存在，见下文。

例子 1：确定型函数关系　假设你是某个社交网站的创立者，这个网站对所有的会员都收取每月 25 美元的会费，而这笔钱是你唯一的经济来源。每个月你都会搜集关于会员数和网站利润的数据并持续了两年的时间。你把这些数据都录在了一个表格当中。从数据的表现形式来说，这些数据可以表示成一个个数据对的形式（用户数，利润值），比如下面就是从数据中摘取的前四个数据点：

$$S = \{(x, y) = (1, 25), (10, 250), (100, 2500), (200, 5000)\}$$

如果你只把这 4 个数据点给你的任何一个朋友看，即便这个朋友根本不知道你在做社交网站以及你是如何收费的（这样的朋友绝交了也罢），他们也能一眼看出数据点中蕴含的数学模型，如果用 y 表示理论，x 表示用户数，则 $y = 25x$。因为其中的模式太过明显，只需要在脑子里面一过就可以迅速得到下面三个结论：

- 两者的关系是线性的；
- 线性关系的强度值是 25；
- 应该是一个确定型的关系，最起码从前 4 个点来看没有例外。

为了再次确认自己的判断，你可以用散点图画出这 4 个点，其中的关系就一目了然了：的确，是一条严格的，确定型的线性关系。参见图 3-1。

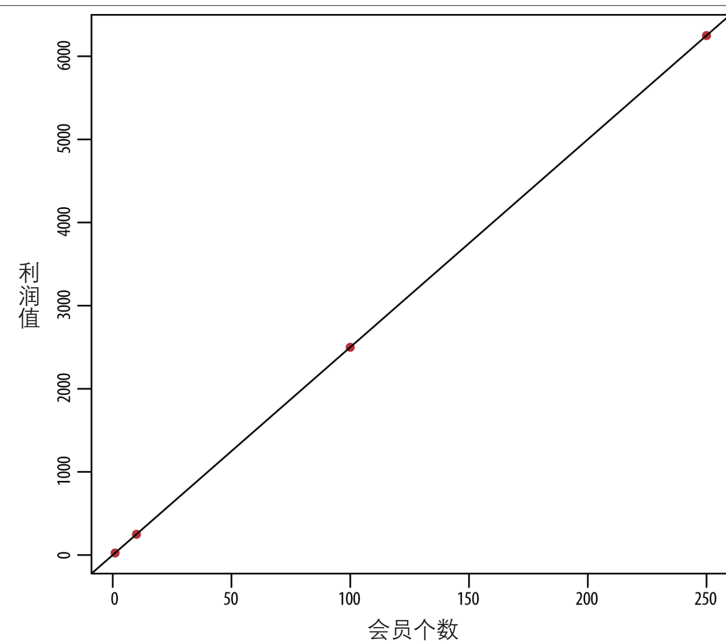


图 3-1：一个严格确定型的线性关系图

例子 2：用户使用数据　假设你有一个用户数据：数据的每一行代表一个用户，数据的

每一列是用户在社交网站上一段时间的某个行为变量。假设数据已经被清理过了，总样本量多达几十万。数据的变量包括“总好友数”“本周新好友数”“总访问量”“总浏览时间”“下载的程序数”“被展示的广告数”“性别”“年龄”等。在探索性分析阶段，你可能只需要从所有的用户中随机抽取 100 个样本即可。为了探索变量之间的相互关系，可以用类似图 3-1 那样散点图的方法。比如这里我们关心的目标变量 Y = 总浏览时间（单位为秒），预测变量 X = 新的好友数。从商业模式的角度来看，如果你的商业模式是卖广告，那么很明显新用户数越多越好，你会承诺给登广告的客户最低的新用户数，这时候就要用到预测模型了，因为你希望能够提前数天或者数周知晓可能的新用户数量。这里我们先把问题简化，看一下两个变量之间的关系。看一眼随机抽取的 100 歌数据，前 6 行是这样的：

```
7   276
3   43
4   82
6   136
10  417
9   269
```

乍一看，不像之前的例子，我们已经很难一眼看出两个变量之间的函数关系了（即使你有读统计学的朋友，他们也未必能一眼看出）。因为实际上还有很多数据，所以这个时候靠拍脑袋是没有用的，不妨把它们的关系散点图画出来，如图 3-2 所示。

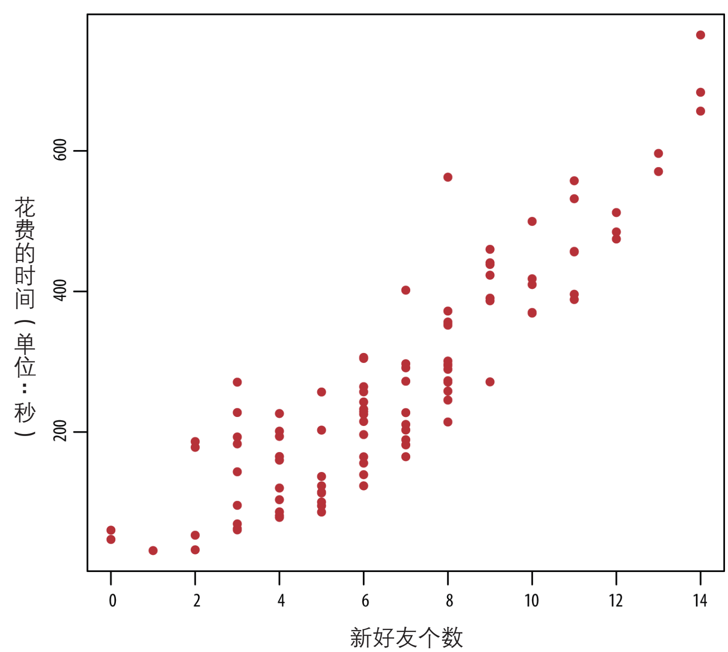


图 3-2：看似线性关系的散点图

从散点图来看，这两个变量之间有着某种线性联系。这个结论是合情合理的，因为如果你的新好友数越多，你很可能会花更多的时间在网站上。但是这里的线性联系不能用一个确定型的函数来表示，因为很明显不是所有的点都在一条直线上。但是即便如此，我们还是可以说，两个变量之间的线性关系是存在的：X 变量的增加同时也带动着 Y 变量的增加，增加的趋势倾向于是一条直线；反之亦然。

小贴士

模型对于数据来说，主要是用来捕捉其中两个方面的信息：第一个是*趋势*（trend），第二个是*变动幅度*（variation）。我们先从趋势说起。

首先我们假设，X 变量和 Y 变量之间的关系确实是线性的，于是我们会尝试在其中画一条直线。

有很多条直线看起来都有可能是不错的选择，我们在其中画了几条，如图 3-3 所示。

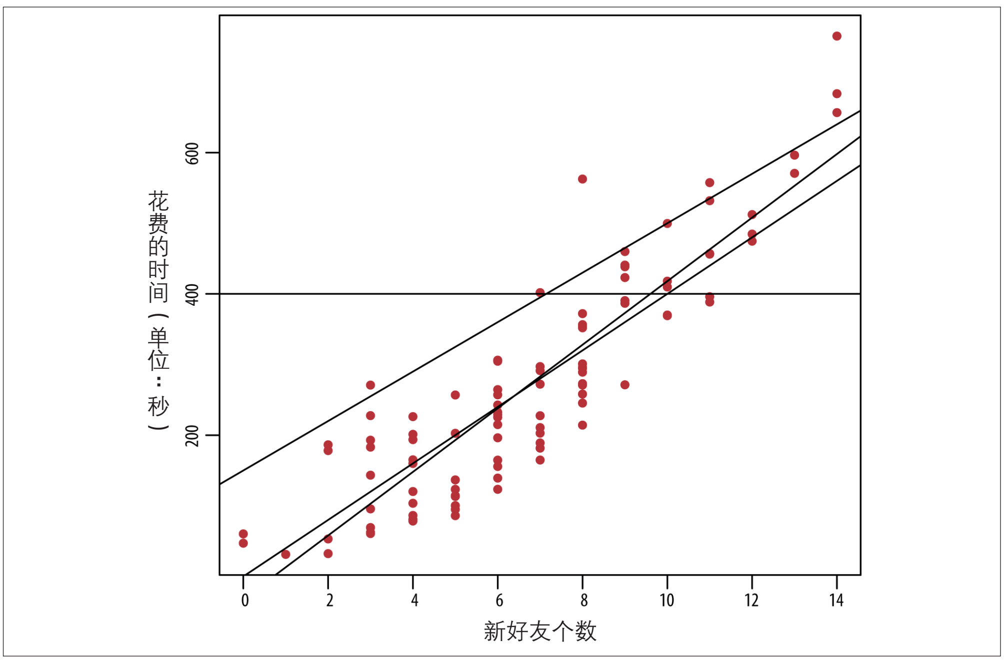


图 3-3：哪条直线最佳呢

这么多直线，我们到底应该选择哪一条呢？

因为我们假设了两个变量之间的关系是线性的，因此模型的形式可以表示为：

$$y = \beta_0 + \beta_1 x$$

那么接下来的工作就是，在给定数据样本 (x_1, y_1)，(x_2, y_2), $\cdots$, (x_n, y_n) 的情况下，确定最佳

的截距（β_0）估计值和斜率（β_1）估计值。

线性模型也可以用矩阵表示：

$$y = x \cdot \beta$$

其中 x 是数据矩阵，β 是参数向量。我们的任务就是，找一条最佳的直线（参数向量估计值）拟合数据。

模型拟合

那么参数向量 β 到底如何估计呢？一个直观的想法是，如果存在一条最佳拟合的直线，那么所有样本数据点到这条的直线的距离应该是所有直线中最小的。

很可能会由很多线看起来都拟合得不错，但是最优的只有一条。可能有很多种定义“最优”的方式，对于线性回归来说，这里的“最优”指的是距离最小化。那么“距离”在这里又是什么意思呢？

我们用图 3-4 为大家讲解一下“距离”的概念。假设数据点的 y 值用 y_i 表示，其在直线上的拟合值（预测值）为 $\hat{y}_i$，那么一个样本点与其拟合值的距离可以定义为两个点在 y 值上的“离差平方”：$(y_i - \hat{y}_i)^2$。所有数据点的距离之和也称作“离差平方和”：$\sum_i (y_i - \hat{y}_i)^2$。最优的那条直线具有最小的“离差平方和”。可以看出，这里距离的定义，也就是“离差平方和”还可以解释为模型的预测误差。这样的估计方法就是著名的最小二乘估计法。

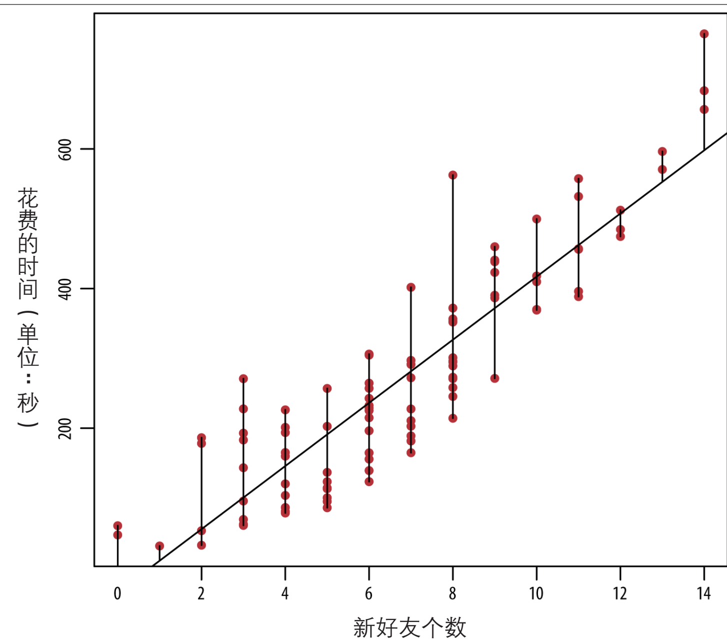


图 3-4：具有最小“离差平方和”的那条最优拟合直线

离差平方和表示为 RSS（Residual Sum of Squares），可以表示为：

$$RSS(\beta) = \sum_i (y_i - \beta x_i)^2$$

i 代表某个数据点。离差平方和是一个有关于 β 的函数，而为了找到最优的 β，我们需要最小化离差平方和。

微积分告诉我们，对 $RSS(\beta)$ 针对 β 求导并令其为 0 即可找到可能的最优解。$RSS(\beta) = (y - \beta x)^t (y - \beta x)$，可以得到：

$$\hat{\beta} = (x^t x)^{-1} x^t y$$

$\hat{\beta}$ 代表 β 的估计值，真实的 β 是无从得知的。在得到 β 估计值的表达式之后，主要将观测数据的值代入即可计算出实际的估计值。

在 R 软件中拟合一个线性模型再简单不过了，假设有一列数据代表因变量 Y，一列数据代表自变量 x，则拟合的 R 代码为：

```
model <- lm(y ~ x)
```

假设真实的数据真像我们之前展示的那样，其前 6 行为：

```
x  y
7  276
3  43
4  82
6  136
10 417
9  269
```

那么用下面的 R 代码：

```
> model <- lm(y ~ x)
> model

Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
     -32.08        45.92

> coefs <- coef(model)
> plot(x, y, pch=20,col="red", xlab="Number new friends",
  ylab="Time spent (seconds)")
> abline(coefs[1],coefs[2])
```

因此，模型最终的最优估计直线为：$\hat{y} = -32.08 + 45.92x$，当然也可以简化为 $\hat{y} = -32 + 46x$，拟合的直线效果见图 3-5 的左半部分。

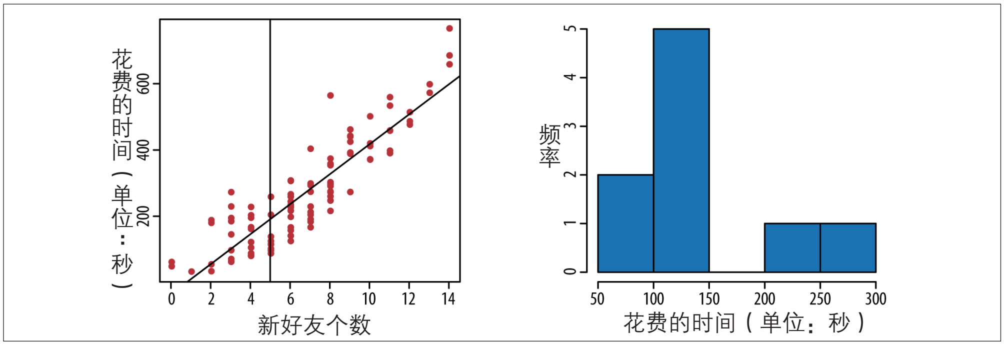


图 3-5：左图是线性回归的最优拟合直线。可以看到，对于任意固定的 x 值，比如 $x = 5$，相应的 Y 值在观测数据层面是可变的。对于 $x = 5$，我们在右图画出了相应 Y 预测值的可能分布

至于到底是否采纳这个线性模型，将它用于数据关系描述和结果预测，这取决于数据科学家自己的判断。如果新数据的 x 值为 5，代表某人的新好友数为 5，那么根据模型，y 的预测值为 $-32.08 + 45.82 \times 5 = 195.7$（秒）。一个自然的问题是，对于得到的预测值，我们有多大的自信认为它会十分接近真实值呢？

这在统计学上叫作置信值的问题，解答它需要将模型的内涵稍作延伸。可以想象，如果用户的新好友数为 5，那么这些用户在网站上花费时间的预测值不可能只是一个定值 195.7 秒，一个合理的情况是这些用户花费的时间都在 195.7 秒附近波动。因此，线性模型得到的预测值只是所有可能预测值的一个总体趋势，而围绕这个趋势的波动性还没有被模型考虑进来。

最小二乘模型的延伸

我们刚刚讨论的是一个简单线性回归模型（一个输出变量，一个预测变量），模型参数的预测采用了最小二乘法来估计 β。在此模型的基础上，我们可以主要从三个方面加以延伸：

(1) 增添一些关于模型误差项的假设；
(2) 增添更多预测变量；
(3) 对预测变量加以变换。

增添关于模型误差项的假设

如果你只是应用线性模型预测给定 x 值情况下的 y 值，那么得到的预测值只是一个确定值，这就忽视了预测必然存在的可变性。图 3-5（右）就很好地说明了这个问题，对于一个给定的 $x = 5$ 的预测情形，y 的值是不定的。为了让模型捕捉数据中的不确定性，可以将模型的形式扩展为：

$$y = \beta_0 + \beta_1 x + \epsilon$$

其中的ϵ是模型中的新加项，也称作“噪声”项，代表数据中不能被模型部分拟合的部分。它也称作*误差项*——ϵ代表模型的实际误差，也就是*实际观测值*与真实回归直线上所得值的差距。真实的回归直线永远是未知的，而你只能通过$\hat{\beta}$估计。

我们通常假设该残差想服从一个均值为 0，方差未知的正态分布，故：

$$\epsilon \sim N(0, \sigma^2)$$

对残差项的正态分布假设有时候是不合理的，比如说当数据具有明显的“厚尾分布”（fat-tailed distribution）特征时，或者模型的主体部分只能拟合数据中的一小部分特征时。在金融数据建模中，这都是经常发生的，因此正态分布的假设很难适用于对金融数据的建模。

但是，这页并不代表我们在金融数据研究中完全摒弃线性回归模型（及其误差项的正态分布假设），只是说金融数据中的厚尾特征为传统的线性回归模型提出了挑战。

在误差项的正态分布假设下，我们可以从条件分布的角度解释线性回归模型。也就是说，对于给定的 x，y 的条件分布是一个正态分布：

$$p(y \mid x) \backsim N(\beta_0 + \beta_1 x, \sigma^2)$$

回到刚才 $x = 5$ 的情形，也就是说，对于所有新好友数为 5 的用户，他们在社交网站上花费的时间服从一个正 e 态分布，该分布的均值为$\beta_0 + \beta_1 * 5$，方差为σ^2。β_0、β_1以及σ^2的值需要从数据中估计。

那么现在的问题是，到底如何拟合这个模型呢？到底如何估计这些参数值呢？

其实，可以从数学上证明，无论误差项的分布形态如何，最小二乘估计都具备同样优良的性质：它是无偏估计量，并且具有最小的估计方差。若想了解该性质以及其中数学证明的细节，我们建议大家 zhao 找一些有关统计推断的书读一读，比如 Casella 和 Berger 合著的《统计推断》（*Statistical Inference*）。

β_0和β_1的估计方法我们已经探讨过了，是基于最小二乘估计的；因此难题是到底如何估计σ^2。基本的想法是用实际的观测误差，也称作残差，去估计实际的误差。实际残差为 :

$$\mathrm{e}_i = y_i - \hat{y}_i = y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i),\ i = 1, \cdots, n$$

σ^2的无偏估计量为：

$$\frac{\sum_i \mathrm{e}_i^2}{n - 2}$$

看到上面的公式，好奇心重的同学肯定会问：为什么上式中的分母是 $n-2$ 呢？这是因为，如果用 $n-2$ 作为分母，而不用 n，那么这个误差估计量在统计学上来说称作一个无偏估计量。$n-2$ 中的 2 时模型中参数的个数。上面提到过的 Casella 和 Berger 合著的《统计推断》一书有详细的背景知识介绍，感兴趣的同学不妨翻看一下。

上面的方差估计量也叫作均方误差（mean squared error），它衡量的是预测值偏离实际观测值的程度。对于预测问题来说，均方误差是被广泛使用的一个误差估计量。对于回归问题来说，它可以作为一个方差估计量，但是对于其他的问题，我们可能不能简单地把它解释为此。接下来我们还会反复提到均方误差的概念。

模型评估标准

我们之前问过类似的问题：对于模型参数的估计值，我们有多大的信心它们是准确的呢？从 R 的模型输出来看，我们可以依赖 p 值和 R 方这两个统计量。在 R 中，在拟合了一个模型之后，如果在控制台中输入 **`summary(model)`**，那么会得到以下输出结果：

```
summary (model)
Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-121.17  -52.63   -9.72   41.54  356.27

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -32.083     16.623   -1.93   0.0565 .
x             45.918      2.141   21.45   <2e-16 ***
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 77.47 on 98 degrees of freedom
Multiple R-squared: 0.8244,    Adjusted R-squared: 0.8226
F-statistic:   460 on 1 and 98 DF,  p-value: < 2.2e-16
```

- *R*方

  $R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$，这是 R 方的定义公式。可以解释为数据中能够被模型所解释的方差占数据总方差的比重。将这个公式与之前均方误差的公式对比，我们会发现，均方误差实际上是 R 方公式中分式的分子部分，分母对应的是数据中的总方差，因此整个分式代表的是数据中未被模型解释的方差的比重。

- *p*值

  在 R 的模型输出结果中，模型参数的估计是在 Estimate 那一列中显示。要想得到每一个参数估计的 p 值，我们要计算 $Pr(>|t|)$。对 p 值的解释我们之前略有提及：我们先设

定了一个原假设（null hypothesis），假设参数值 β 为 0。对于每一个 β，p 值代表的是在原假设的基础之上我们可以得到观测数据的概率，也就是在原假设的基础上得到相应 t 统计量值的概率。这也就意味着如果 p 值很小，那么在相应的原假设下得到观测数据的概率就很小，因此原假设很可能是不正确的。也就是说，β 应该显著得不为 0。

- 交叉验证

  还有另外一种评估模型的方法，它大概要遵循这样的程序：分割数据，80% 用作训练数据集，20% 用作测试。在训练数据及上拟合模型并估计模型的参数，并在测试数据集上计算出模型的均方误差，并与训练数据集上模型的均方误差进行比较。如果两种均方误差的差异很小，那么可以认为模型具有不错的扩展性，其过拟合的风险较小。我们也建议大家尝试改变一下训练数据集的大小，看看两者之间有何联合变动关系，这样的模型验证过程叫作“交叉验证法”，如图 3-6 所示。

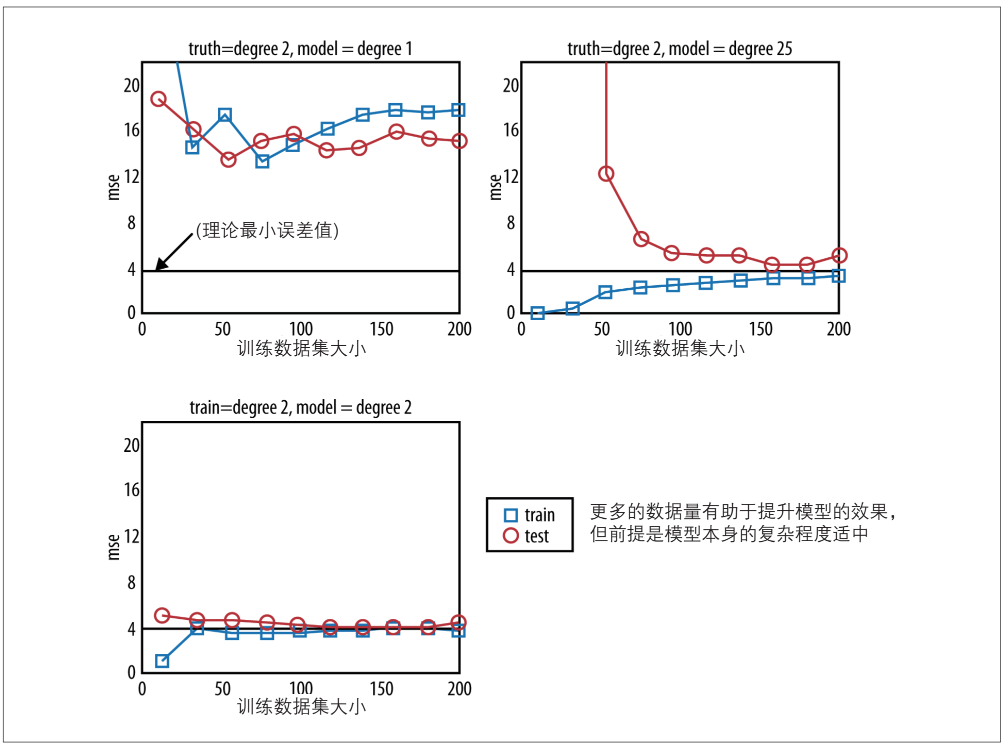


图 3-6：在训练数据集和测试数据集上比较均方误差。该图截自 Nando de Freitas 教授的课件。这里，由于数据是从一个一直的分布模型中模拟得来的，因此真实的模型误差是事先知道的

其他类型的模型误差测度

均方误差属于“损失函数”的一种，在线性回归中均方误差是最常使用的误差指标，它能够很好的测度模型的拟合效果。它的另外一项优点就是，如果假设误差项是正态分布的，那么模型的估计可以完全依赖最大似然函数法。其他类型的模型误差测度还有很多，比如

基于绝对值误差。人们甚至可以根据研究的目的和个人的偏好设计和使用相应的误差测度指标。但从目前来看，我们还是觉得均方误差是个不错的选择。

增加预测变量 我们刚才讨论的是最简单的线性回归模型：一个自变量以及一个因变量。在此模型的基础之上，我们可以增加预测变量（自变量）的个数，从而得到*多元回归模型*：

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \epsilon$$

之前的模型表达式在这里也同样适用，因为我们之前用了矩阵表达式，增加一个变量无非是在矩阵上增加一列而已，模型的表达形式不会改变。还以刚才的例子来说，对于预测人们在网站上所花去的时间，人们的年龄和性别也可以作为预测变量。至于在线性回归模型中到底应该选用哪些变量，我们将在 7.4 节详细介绍。对于上面的具有三个预测变量的模型，在 R 中的模型拟合代码为：

```
model <- lm(y ~ x_1 + x_2 + x_3)
```

若要在模型中添加一个交叉变动项也很简单：

```
model <- lm(y ~ x_1 + x_2 + x_3 + x_2*x_3)
```

在确定到底使用哪些预测变量时，散点图通常是一个很好的工具。我们可以画出预测变量与每一个自变量之间的散点图，以及自变量之间的散点图，已找到或确认可能有用的预测变量和变量之间的交叉变动关系。基于预测变量的每一个值，画出因变量的条件分布（$y \mid x$）直方图也会有所帮助。在选定了预测变量之后，同简单线性回归模型一样，我们仍然可以使用 R^2、p 值和交叉验证的方法评价模型的质量。

变换 刚才的模型假设了 y 与 x 之间的线性关系，然而为什么我们不能使用 x 的高阶项呢，这样我们就可以得到一个类似下式的多项式回归模型：

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$

我们知道，多项式回归模型不是*线性回归模型*。但是如果把 x 的高阶项看成新的变量，上式的表达形式仍然保持着“线性”的模样。也就是说，我们可以把多项式回归模型看成线性模型，前提是把变量做一下适当的变换。比如，我们可以假设 $z = x^2$，$w = x^3$，这样上面的模型就是一个基于三个预测变量 (x, z, w) 的线性回归模型了。这样的变量变换在线性回归中经常会用到，其他的变换方法还包括取对数、离散化、二元化等。

但是，是否对一个变量进行变换是建立在事实基础之上的，比如说在预测人们在网上花费的时间这一问题上，其中一个预测变量就是好友的个数。通过散点图，如果我们发现好友的个数与花费时间的关系呈现明显的曲线特征，而非直线，那么我们就应该考虑对“好友的个数”这个变量做适当的变换，以满足线性回归模型的“线性”要求。

我们现在所讨论的各种有关模型形式和变量变换的问题，是建模工作者必须要直面的一个

最重要的挑战之一：那就是你永远不知道真理是什么。很可能真实的模型是二次型的，而你最后还是使用了线性模型。有许多的工具可以帮助我们尽量客观地确定模型的形式，但你永远不能 100% 地肯定你的模型是正确的。数据量越多往往对于建模越有帮助，我们可以无限地接近真理，但永远无法到达那里。

回顾一下

现在让我们回顾一下我们在构建模型和拟合模型的时候做过的诸多假设：

- 线性假设；
- 误差项是正态分布的，并且均值为 0；
- 误差项是相互独立的；
- 误差项具有恒定的条件方差；
- 预测变量都是有用的。

我们应用线性回归模型，主要基于以下两个目的：

- 用作预测，在知道自变量值的情况下，预测因变量的可能值；
- 用作变量关系的解释，比如确定两个变量之间可能的相关关系或者联合变动关系等。

练习

我们可以在 R 中用模拟的方法帮助理解和探索这些新概念。模拟的好处在于，你永远知道数据是如何产生的，因此也就知道真实的模型是什么样子。换句话说，你就是“上帝”，真理是掌握在你自己手中的。因此，你可以评判一个模型的效果到底好还是不好。

在模拟的数据上确认完模型的有效性之后，就可以把目的转移到真实的数据上了。接下来我们会展示如何模拟出一个数据集，以及如何利用模拟数据集帮助我们探索和理解模型：

```
# 模拟一个虚拟数据集
x_1 <- rnorm(1000,5,7) # 从一个均值为 5，标准差为 7
                       # 的正态分布中随机模拟 1000 个值
hist(x_1, col="grey")  # 画出 p(x)
true_error <- rnorm(1000,0,2)
true_beta_0 <- 1.1
true_beta_1 <- -8.2
y <- true_beta_0 + true_beta_1*x_1 + true_error
hist(y) # 画出 p(y)
plot(x_1,y, pch=20,col="red") # 画出 p(x,y)
```

(1) 在模拟数据上拟合线性回归模型，比较参数 β 的估计值与真实值的差距。

(2) 模拟一个新的服从伽马分布的变量 x_2，并构建一个新的 y 变量：$y = x_1 + x_2$。接下来可以做很多事，我们可以拟合一个只有 x_1, 或者只有 x_2 的模型，再拟合一个包含两个变量的模型。随后，我们可以在不同样本量大小的数据集上拟合模型，并可以通过交叉验证的方法看模型的均方误差在训练数据集和测试数据集上的表现。

(3) 在模型中加入新的预测变量 z，$z = x_1^2$。比较模型在加入该变量前后的拟合效果有何不同。通过交叉验证和改变样本量的方式，比较模型在训练数据集和测试数据集上的不同表现。

(4) 还有很多东西可以玩：(a) 改变真实的参数值；(b) 改变模型误差项的分布类型；(c) 在模型中加入更多具有不同分布类型的预测变量。（在 R 中，`rnorm()` 函数的作用是从某个正态分布中生成一些随机值。顾名思义，`rbinom()` 函数是从二项分布中产生随机值。可供尝试的分布类型非常多，大家可以多尝试几种。）

(5) 画出所有变量之间关系的散点图，以及单个变量的直方图。

3.2.2 *k* 近邻模型（*k*-NN）

k 近邻模型是一个分类算法，在给定一个已经做好分类的数据集之后，*k* 近邻可以学习其中的分类信息，并可以自动地给未来没有分类的数据分好类。

分类是建模的基本问题之一，我们可能想把数据科学家分成两类：“性感类”和“不性感类”，或者把人分成两类：“高信用度”和“低信用度”；酒店则可能按星级分成 5 类：“五星级”“四星级”“三星级”“二星级”和“一星级”。对于酒店，也许我们可以多加一类，将那些非常差，连“一星级”都不够格的酒店分类为“零星级”。对于患者，我们可能希望将他们分为“高患癌风险病人”和“低患癌风险病人”两类。分类的任务在我们日常生活中可以说无处不见。

现在让我们停下来思考一下，对于分类问题我们可以应用线性回归模型吗？

答案是：不一定，也不是没有可能。从模型形式上来说，线性模型的输出值是连续性的实数值，而分类模型的任务要求是得到分类型的模型输出结果。从这一点上来说，线性模型是不适用于分类问题的。

但是，如果我们换一个思路，问题还是可以解决的。这里要用到“阈值”的概念。也就是将连续性数值离散化。比如，如果我们想根据人们的年纪和实际收入预测他的信用值，那么我们可以选取 700 作为阈值，如果其信用预测值高于 700，则将其列为“高信用度”类。如果其信用预测值低于 700，则将其归类为“低信用度”类。这里我们用一个阈值将一个连续性变量转换成了一个二元变量，我们当然可以使用更多的阈值将预测性变量离散化成具有更多类别的变量。比如，可以用 4 个阈值将信用度分成 5 个类别：极低信用度、低信用度、中级信用度、高信用度和极高信用度。

然而，并不是所有的变量都像信用度一样，可以被阈值分为几个严格不重叠的类别。有些类别可能是模棱两可的，比如一个人的政治倾向，他可能是“民主党”，也可能是“共和党“，或者可能是政治中立的。这样的具有概率特性的分类变量，我们该如何分析呢？

k 近邻的主要想法是，根据属性值的相似度找到某个对象的相似对象们，并让其相似对象

们一起“投票”决定该对象到底应该属于哪一类。如果有某两个或者更多个的类别投票数相同，那么就从这些类别中随机挑选一个作为该对象的类别值。

让我们举一个例子。假设一个人会对自己看过的每部电影都打上“好看”或者“不好看”的标签，现在根据这些历史标签信息，如果给定一部新的电影叫作《狂野的数据》，我们可以用 *k* 近邻的方法预测此人是否会觉得这部电影好看。方法首先是要确定电影的属性值，包括电影的长度、风格、性爱场景数、奥斯卡最佳演员的个数以及电影的拍摄预算等。根据这些属性，我们可以找到与《狂野的数据》最相似的 *k* 部电影，并记录下这 *k* 部电影的标签信息。如果这些电影总体来看包括更多的是“好看”的标签（这个过程，叫作“投票”），那么我们也自然会预测《狂野的数据》这部电影会得到此人的青睐。

k 近邻方法需要解决两个核心问题：一是如何根据属性定义个体之间的相似性或者紧密程度。一旦有了明确的定义，我们就可以把某个带预测个体的“近邻”们找出来，让它们投票决定该个体的类别属性。

这就自然引申出了第二个问题：到底如何才能确定一个最优的 *k* 值呢？对于数据科学家来说，*k* 值的确定十分关键。接下来我们会详细讨论解决方法。

首先让我们看一个现实生活中的例子。

信用评分实例

设想手边有一个关于人们信用评分的数据集，包含的变量有人们的年龄、收入以及信用评分的等级。信用等级变量只包含两个类别，分别是“高信用度”（high）和“低信用度”（low）。给定这个数据，我们想据此预测一个新人的信用等级。

下面是这个数据集的前几行，收入在第二列，其单位是千美元：

```
age  income  credit
69      3     low
66     57     low
49     79     low
49     17     low
58     26    high
44     71    high
```

图 3-7 是一个特别的散点图，横轴是年龄变量，总重是收入变量，实心点代表高信用度，虚心点代表低信用度。

现在有一个预测问题：根据刚才的数据集的信息，如果告诉你现在有位新人，他的年龄是 57 岁，收入是 37 000 美元，那么他的信用等级应该是什么呢？或者说，他的信用等级更可能是高还是低呢？如图 3-8 所示，问号的位置代表新人的两个属性（年龄和收入）的位置，根据他周边的标签信息，*k* 近邻方法可以告诉我们他的信用等级更应该是高还是低。

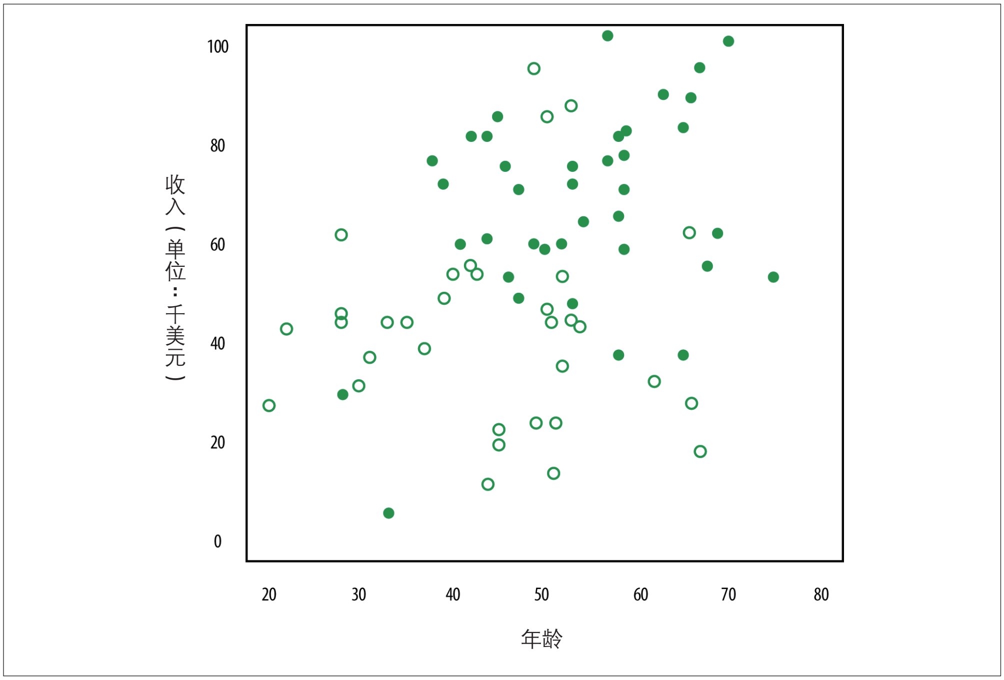


图 3-7：信用等级关于年龄和收入变量的散点图

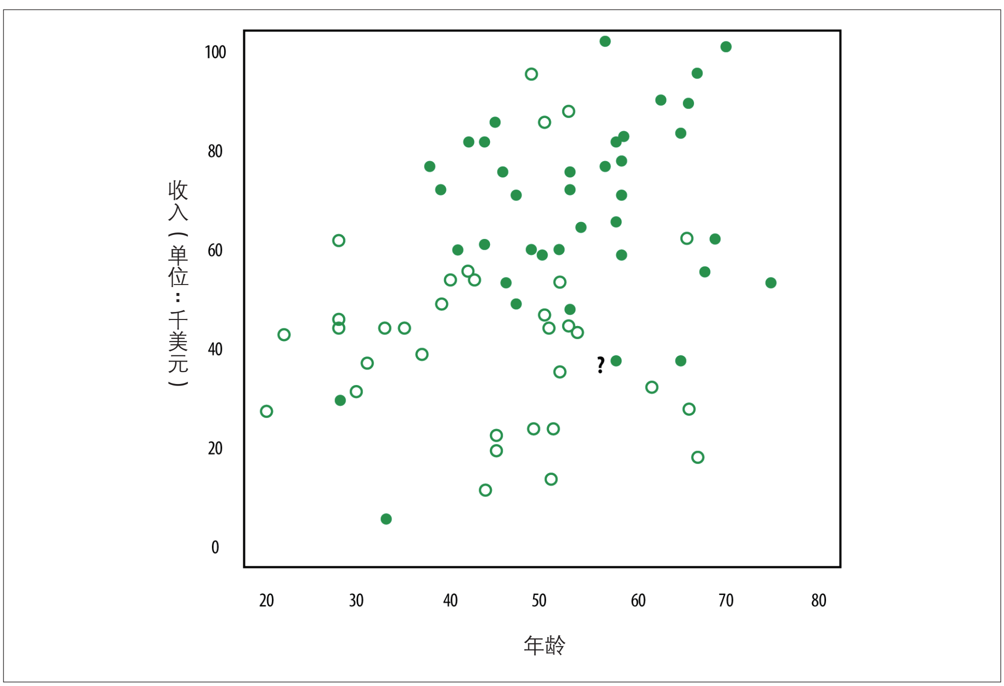


图 3-8 ：这个新人的信用等级是高是低呢

下面我们就列出 k 近邻方法工作的详细步骤。

(1) 首先确定相似性定义，这通常也叫作“距离”。
(2) 将数据分割成训练数据和测试数据集。
(3) 选择一个模型评价标准。（对于分类问题来说，误分率是个不错的选择，我们会在稍后章节详细讨论。）
(4) 选择不同的 k 值，并应用 k 近邻模型，看哪个模型的效果最好。
(5) 基于选定的模型评价标准，选出最优的 k 值。
(6) 选定 k 之后，就可以做样本外测试了。我们刚才提到的一个新人就是一个样本外的例子。

相似性/距离测度

相似性测度的选择在很大程度上要取决于问题的背景和数据本身的特征。举个例子来说，对于社交网络数据来说，如何衡量两人之间的“相似性”呢？一个通常的做法是用两人之间共同好友的个数来衡量。而这样的测度方式放在别的数据上可能就不太合适了。

对于我们刚才讨论的问题来说，如果变量取值大致都在一个水平上，那么二维平面上的欧几里得距离是个不错的选择。当然，我们很难假设所有变量的取值都在一个水平上，很多时候，变量的取值范围会差别很大。

注意：前方有坑！

对于建模来说，变量的取值水平是个很重要的问题，如果处理得不好，很可能会影响整个模型的效果。

让我们来看一个例子：年龄通常的取值单位是年，收入是美元，而信用评分的取值往往是由权威部门规定的——就像 SAT 分数一样。因此一个人的观测数据可以用一个三元向量表示，如（25, 54 000, 700）和（35, 76 000, 730）。 因为收入的取值水平最高，因此在计算距离的时候，相似性的测度很可能会被收入这一个变量所主导，而其他两个变量很难在该测度中体现出来。这就是变量取值水平的问题，我们总是希望变量的取值都大致在同一个水平上。

然而，如果收入是用“千美元”来度量，那么相应的三元变量会变为（25, 54, 700）和（35, 76, 730）。如此一变，三个变量的取值就大概在同一水平上了。

总而言之，变量的计量方式以及变量之间距离的定义方式，在统计学中我们叫作“先验信息”，他们都可能会影响到模型最终的效果。

欧几里得距离又叫欧氏距离，对于在实数轴上取值以及能够在平面或者多维空间中描绘出来的变量来说，它是一个不错的距离亮度。

- 余弦度（Cosine Similarity）

  余弦度同样可以用于亮度两个实数向量$\vec{x}$和$\vec{y}$之间的相似程度，而且余弦度是一个介于 –1 和 1 之间的数。如果取值为 –1 则代表完全不同，1 代表完全一样，而 0 则代表两个向量之间相互独立。回顾一下余弦度的定义：$\cos(\vec{x},\vec{y}) = \frac{\vec{x},\vec{y}}{\| \vec{x} \| \vec{y} \|}$。

- Jaccard距离或相似度

  Jaccard 距离通常用作衡量两个集合之间的相似度——比如说 Cathy 的朋友集合表示为 $\boldsymbol{A} = \{\text{Kahn, Mark, Laura, } \cdots\}$，Rachel 的朋友集合表示为 $\boldsymbol{B} = \{\text{Mladen, Kahn, Mark, } \cdots\}$，那么这两个朋友集合的 Jaccard 相似度为$J(\boldsymbol{A},\boldsymbol{B}) = \frac{|\boldsymbol{A} \cap \boldsymbol{B}|}{|\boldsymbol{A} \cup \boldsymbol{B}|}$。

- Mahalanobis距离

  也称作马氏距离，同样适用于两个实数向量。相比于欧氏距离，马氏距离考虑了两个变量之间的相关关系，并且不用担心变量取值水平的问题。马氏距离的定义为：$\mathrm{d}(\vec{x},\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T \boldsymbol{S}^{-1}(\vec{x}-\vec{y})}$，其中 $\boldsymbol{S}$ 是两个向量间的协方差矩阵。

- Hamming 距离

  Hamming 距离主要用来衡量两个字符串，字组或者具有相同长度的 DNA 序列之间的相似程度。比如，单词 olive 和 ocean 的 Hammning 距离为 4，因为除了第一个字母 o 相同之外，这两个单词包含的其他四个字母都不相同。同理，单词 shoe 和 hose 之间的 Hamming 距离是 3，这是因为除了最后一个字母 e 相同之外，其他三个字母在对应位置上均不相同。因此，Hamming 距离的计算方式其实十分简单，按照位置顺序的比对两个单词字母之间是否相同，每个位置上字母的不同，相应 Hamming 距离的值都增加 1。

- Manhattan距离

  Manhattan 距离（曼哈顿距离）也用来测度两个 k 维实数向量之间的距离。之所以叫作 Manhattan 距离，是因为要把这个距离想象成城市中两点之间，如果用计程车行走的路线来算的话，应该为多长。（在城市中行走或者行车需要遵循一定的路线规定，譬如大型建筑要绕行，不能穿墙而过。）曼哈顿距离的定义为：$\mathrm{d}(\vec{x},\vec{y}) = \sum_{i}^{k} |x_i - y_i|$，$i$ 代表向量中元素的位置。

除了上述的这些距离测度指标，当然还有很多别的选择。至于到底选择什么样的指标，这要取决于研究的问题和数据本身。当你觉得困惑的时候，不妨先到谷歌上搜索一下，看看别人是怎么用的。

有一种情形会使得问题变得更加复杂，那就是当数据有很多不同类型变量的时候。比如说，在电影评分数据库中，有些变量是数值型的（比如电影预算和演员数量等），而有些是分类型的变量（比如电影的风格）。在选择距离测度的时候，我们还要根据变量类型的不同而有所变通。但无论如何，距离的测度选择都具有相当的自由和主观性。

我们甚至可以自己定于距离的测度方式，比如对于电影评分数据来说，我们可以定义，如果两个电影的风格相同，那么它们之间的距离值增加为 0，但是如果风格不同，则距离值增加为 10。之所以选择 10 这个数据作为单位增加值。至于到底选择 10，50 还是更大的值作为单位增加值要取决于其他变量的取值水平而定。如果其他变量的取值水平较大，那么很可能 50 会更好一点。

无论做何选择，我们总是希望这个选择在可能的选择中是最优的，虽然很难做的，但也还是值得一试。从模型验证的角度来说，我们尝试很多的选择，并通过模型验证的方式确认哪一个会是最优的。在刚才的选择中，10 相对于 k 近邻算法本身是第二个需要确定的数值（k 是第一个，也是最重要的），我们可以把它作为模型的参数最优化，也可以作为建模之前已经掌握的先验信息。至于到底应该怎么看待它，也同样取决于这个问题本身，研究人员自己看待这个问题的角度以及数据本身的特点等。

训练和测试数据集

对于所有的机器学习算法来说，将数据集分成训练数据及和测试数据集是最为通用的做法。数据集用来“训练”模型，而测试数据用来独立的测试“训练好”的模型在实际问题上的真实表现。

对于 k 近邻模型来说，训练阶段的数据是包含因变量（信用等级）的标签值的。在模型测试阶段，我们假装不知道因变量的真实标签值，并用在训练阶段得到的 k 近邻模型预测这些标签值。

要想实现测试的目的，我们必须从原始数据中保留一些数据出来，这些数据不用于模型训练而只用于测试。通常来说，我们会随机抽取 20% 的数据出来作为测试用数据集。

以下的 R 代码可供参考：

```
> head(data)
  age income credit
1  69      3    low
2  66     57    low
3  49     79    low
4  49     17    low
5  58     26   high
6  44     71   high

n.points <- 1000 # 数据集中的行数
sampling.rate <- 0.8

# 我们需要测试数据集中的行数去计算误分率
num.test.set.labels <- n.points * (1 - sampling.rate)

# 在所有的数据行中，随机的选取一部分用作训练数据集
training <- sample(1:n.points, sampling.rate * n.points,
                   replace=FALSE)
```

```
train <- subset(data[training, ], select = c(Age, Income))
# 这些行数就代表训练数据集的规模
# 剩下没有被随机选中的数据行就代表了测试数据集
testing <- setdiff(1:n.points, training)
# 没有被选中的行数就代表了测试数据集的规模
test <- subset(data[testing, ], select = c(Age, Income))

cl <- data$Credit[training]
# 这些是训练数据集的标签值
true.labels <- data$Credit[testing]
# 测试数据集的标签值暂且保留
```

选择一个模型评价标准

到底应该如何评判你所选用的模型（及其参数值）的效果呢？

就像我们之前反复说的，模型的评价是非常灵活和主观的，并没有普适的标准。对于某个实际问题，你可能觉得高误分率的严重性要大于其他的误差测度，或者你会觉得伪阴性的严重性要大于伪阳性。因此，在确定一个模型的评价标准时，你可能要和某个问题领域的专家好好交流一番。

举例来说，如果分类模型的对象是人们是否患癌，那么模型的伪阴性当然越小越好（伪阴性指的是某人实际患癌，但是模型却告诉我们他没有患癌）。在确定合理的模型评价标准之前，我们最好与癌症领域的专家医生做好交流沟通，以更深入地了解手中数据所涉及的实际问题背景。

但是对于伪阴性，有一个极端的问题是：如果你想确保伪阴性为 0，你可以把所有的病人都分类为癌症患者。这当然也不合理，因为相应的伪阳性率会增加。在分类问题中，这叫作敏感度（sensitivity）与“特异度”（specificity）的权衡。敏感度指的是所有的患癌病人实际被诊断为患癌的概率，而特异度指的是未患癌的病人被诊断为未患癌的概率。理想的模型需要具备高敏感度和高特异性。

关于敏感度和特异度的其他说法

敏感度又叫作“真阳性率”（true positive rate）或者召回率（recall)。来自不同学术研究领域的研究人员会使用不同的术语，但是它们都代表的一个意思。特异度也叫“真阴性率”（true negative rate）。既然有“真阳性率”和“真阴性率”，当然也有“伪阳性率”和“假阴性率”，但是后两个术语没有其他的表述形式。

我们也可以用“精确度”（precision）作为模型的评价标准，关于精确度我们会在第 5 章详细介绍。之所以有些术语在不同的领域有不同的表述形式，是因为关于分类的问题的研究在统计学，机器学习等领域是独自发展的，针对同样的问题也就自然的出现了一些不同表述。比如说，“精确度”和“召回率”这样的术语就来自于信息检索领域，但要注意的是，

“精确度”不同于“特异度”。

还有一种模型评价标准叫作“准确度”，指的是所有被正确分类的标签数占总标签数的比例。因此，误分率 = 1 – 准确度。最小化模型的“误分率”就等同于最大化模型的“准确度”。

小结

在介绍完距离的定义和模型的评价标准问题之后，模型的前期准备工作就基本完成了。

对于测试数据集中的每个人，你都先假装不知道他们的真实信用等级。用 *k* 近邻方法找到与每个人最相似的三个人的信用等级，并用“投票”的方式确认此人的信用等级。在预测完测试数据集中所有人的信用等级之后，与他们真实的信用等级做比较，并计算出测试数据集上的误分率，它就代表该 *k* 近邻模型的实际预测效果。在 R 中，只需要一行代码就可以完成这些工作：

```
knn (train, test, cl, k=3)
```

k 的选择

我们已经说过，*k* 对于 *k* 近邻模型来说是最为重要的参数，至于如何选择一个合适的（或者最优的）的 *k* 值，这要求你对数据本身的背景有深刻的理解，并且还需要通过不断地尝试（基于选定的模型评价标准）以确定一个合适的 *k* 值。

> **二元分类**
>
> 刚才的信用等级分类是一个二元分类问题，因为待分类的变量“信用等级”只有两个类别“高信用度”和“低信用度”。对于二元分类情况，我们建议选择一个奇数 *k* 值。原因是，奇数 *k* 值在“投票”的过程中总会出现一个“多数”类。当然，你也可以选择一个偶数 *k* 值，如果需要“投票”出现平手的情况，只需要随机挑选一个类即可。

```
# 我们将迭代 20 个 k 值，并且看哪个 k 值对应的误分率最小
for (k in 1:20) {
   print(k)
   predicted.labels <- knn(train, test, cl, k)
   # 这里使用了 R 中的 knn() 函数
   num.incorrect.labels <- sum(predicted.labels != true.labels)
   misclassification.rate <- num.incorrect.labels /
                             num.test.set.labels
      print(misclassfication.rate)
   }
```

下面是上面一段代码[4]的输出值：

```
k  misclassification.rate
1,  0.28
2,  0.315
```

注 4：该段代码尝试了 20 个不同的 *k* 值，并在测试数据集上评价每一个 *k* 值的实际预测效果。

```
3， 0.26
4， 0.255
5， 0.23
6， 0.26
7， 0.25
8， 0.25
9， 0.235
10，0.24
```

从测试的输出结果来看，$k = 5$ 是最优的选择，因为其对应的误分率最低。把 $k = 5$ 的 k 近邻模型应用到之前提到的新人上，该人年龄是 57 岁，收入为 37 000 美元。在 R 中，敲入以下代码：

```
> test <- c(57, 37)
> knn(train, test, cl, k = 5)
[1] low
```

k 近邻模型告诉我们，应用 $k = 5$，该人的信用度为“low”（低信用度）。

> **k 近邻模型中的测试数据集**
>
> 我们刚才使用了两次 knn() 函数，虽然我们都称为“测试”但是意义却不同。第一种意义上的“测试”，主要是用来衡量模型效果的。因为在第一个测试集上，我们其实是知道真实的标签值的，我们只是在把测试数据集指定给 R 的时候假装不知道而已。而第二种测试数据集（比如那个新人的数据）是真正意思上的“样本外测试集”，也就是说，我们真的不知道该人的信用等级如何，而想利用 k 近邻模型预测得知。可以看到，对于 R 来说，这两种意义上的“测试”从代码形式上来看是完全一样的，R 并不知道你所要做的“测试”是何种意思上的测试。

模型有哪些假设

上一章我们讨论了模型和模型的假设问题。那么对于 k 近邻模型来说，有哪些模型假设呢？

k 近邻从模型形式上来看是一个非参数（nonparamatric）的方法，也就是说，对于数据的生成过程和数据的分布没有任何假设，也没有任何需要估计的参数。但即便作为一个非参数的方法，还是有一些隐含的假设，如下。

- 在数据的特征空间中可以定义某种意义的“距离”。
- 训练数据集中的因变量已经做好标签。
- k 值的选择，需要你来决定。
- 我们选择和观测的特征变量对于预测因变量的标签值有所帮助，也就是说，这些特征变量与因变量是有联系的。但很可能某些特征变量与因变量是没有联系的。至于这些特征变量的预测效果到底如何，后期的模型评价自然会告诉我们。在建模的过程中，基于模

型评价的反馈，你可能会考虑添加某些特征变量，或者剔除某些特征变量。这些都叫作模型调优，好的模型应该经过仔细地调优。但在调整的过程中，也要注意避免过拟合的问题。

线性回归模型和 k 近邻模型都属于“监督性学习算法”，也就是说，预测变量的观测值 x 和待预测变量的观测值 y 都是已知的，学习的目的是找到 x 产生 y 的函数。接下来我们要介绍的算法属于“非监督性学习算法”，它适用于 y 的观测值未知的数据。

3.2.3 *k* 均值算法

之前介绍的算法都属于监督性学习算法，它适用于 y 的观测值（对于分类问题来说就是待预测变量的标签值）已知的情况。在这个背景下，我们总是希望模型对 y 的预测越精确越好，这也是为什么在监督性学习中存在模型评价标准的原因。

“非监督性学习算法”的典型代表是聚类分析，在聚类分析中 y 的真实标签值是未知的。k 均值是我们要接触到的第一个“非监督学习算法”。

假设我们有一些用户层面的观测数据，比如 Google+ 的数据，某些调查数据，医学实验数据或者 SAT 分数的数据。

每个用户都可以表示成某些特征变量的组合，比如年龄、性别和收入等。假设下面一行就是用户数据的所有特征变量：

```
age（年龄）gender（性别）income（收入）state（所在州）household size（住户人数）
```

我们的任务是根据用户变量信息的不同把用户分成几组。这个任务也有很多其他的名字，比如*分层*、*分组*、*聚类*等。但是它们都指的是一个意思，那就是根据用户特征的信息把相似的用户组合起来形成类别。

聚类的目的有很多，下面我们举几个例子说明。

- 市场营销中经常需要根据用户的不同特征，提供给用户不用的产品或者服务。比如，打印机公司肯定只想把墨盒的营销广告展示给已经有打印机的用户。
- 也许你手中的模型只适用于某种类型的群体数据，或者你想对不同的群体数据应用不同的模型。
- 统计学中的*分层模型*就内含了聚类的想法：比如在家庭消费调查数据中，分层模型会根据地理位置信息的不同，对不同的区域采用不同的模型形式。

为了展示为什么聚类分析是有用的，试想一下在建模过程中，我们经常需要对某些特征变量（属性变量）进行离散化或者分块化（分组）。

比如年龄变量可以分块：20~24 周岁、25~30 周岁，等等。收入变量也同样可以分块，而像“所在城市”或者“所在州”这样的变量因为已经是分类变量，就不需要再做分块了。但是，如果一个分类变量的类别过多，分块仍然是必要的，当然这要取决于你所采用的模型以及样本数据量的大小。比如，“所在州”变量可以再分块“所在区域”变量，该变量中包含较少的类别：东西部、中部，等等。

假设年龄被分成了 10 组，性别是两组，并且该被分块的变量都已经完成了分块，那么我们可能会得到 $10 \times 2 \times 50 \times 10 \times 3 = 30\,000$ 种不同的组合。这对于建模来说，已经是很大规模的分组了。

由于我们有 5 个分组过的变量，试想一下，在一个五维空间中，每个变量都对应一个坐标轴：因此，有性别轴、收入轴等。每个分组都对应相应的坐标轴点。这样做的结果是这些坐标轴交织形成的网将包含每一个可能的属性变量的组合值。

每个数据点都在这个空间中对应坐标轴系统内的某一个组合点。这样看来，30 000 个不同的组合实在是过多了。比如，从市场营销的角度来说，没有任何一家公司想指定 30 000 种不同的营销策略。

也许这个时候，也才会感到无助而尝试寻求算法的帮助了，尤其是在这种情形下，你想在建模之前进行分组，却又不想得到上万个不同的组别。这就是 k 均值聚类可以做的事情：k 就是这个聚类算法最后所得到的类别数。[5]

二维的问题

刚刚的五维问题可能很难在脑海中有画面感，现在我们讨论一个更简单的二维问题。这里假设数据是有关广告点击率的，我们知道变量时两个，其中 x 是用户被展示的广告数，另外一个变量 y 是用户点击广告的次数。

图 3-9 用散点图展示了数据的大致分布形态。

注 5：k 通常是一个小于 10 的数，对于市场营销来说，10 个类别肯定要比 30 000 个要容易操作。

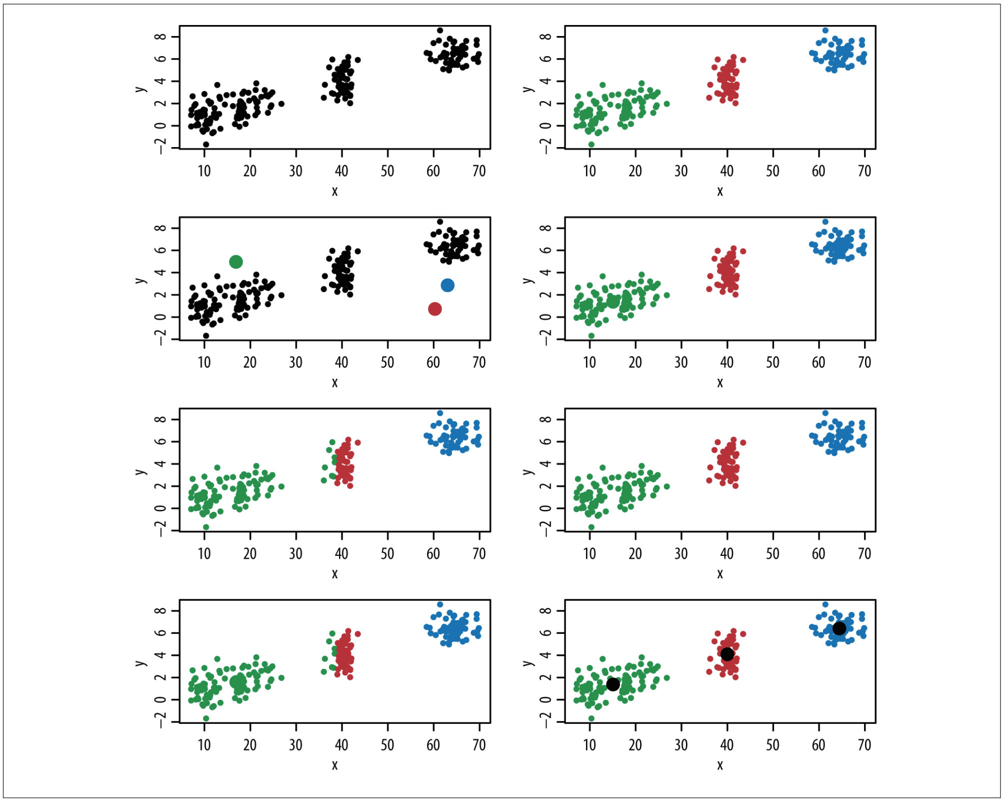


图 3-9：二维空间上的聚类过程，先看左半边从上往下，再看右半边从上往下（另见彩插图 3-9）

从图 3-9 的左上角的散点图可以看出，数据自然地聚成了几类。对于二维空间的数据来说，如果数据点不多，我们很容易就可以通过肉眼判断出数据大概聚成了多少类。当数据维数增加或者数据量变大之后，就不可能依赖于肉眼判断了，你需要一个类似于 *k* 均值聚类的算法帮助你判断数据中到底有多少类别。*k* 均值聚类可以在 *d* 维空间中找到聚类，其中 *d* 是数据空间的维度，也是每个数据点所具有的特征变量的个数。

图 3-9 展示了聚类的整个过程，可以总结为以下四步。

(1) 首先，在数据空间中随机挑选 *k* 个点（叫作中心点，centroid）。这也叫 *k* 均值聚类的初始化操作，要尽量让中心店靠近数据点，并且各个中心点之间要明显不同。

(2) 将数据点分类到离它最近的中心点。

(3) 重新计算中心点的位置。

(4) 重复上述两部直到数据中心点的位置不再变动，或者变动幅度很小为止。

至于如何确定算法已经迭代完毕也没有绝对的标准。甚至对于 *k* 的选择也没有标准，毕竟这是一个非监督性的学习方法，*k* 的大小要通过不断地尝试，以确定一个较为合适的值。

这是一个典型的非监督性学习的例子，因为所有的数据点都没有事先做好的标签类别值，都是通过算法来发现。

k 均值聚类算法有一些已知的缺点。

- *k* 的选择是颇具艺术性的。当然，*k* 的取值也是有上下限的，需要满足 $1 \leqslant k \leqslant n$，其中 *n* 是数据的样本量。
- 收敛性问题：可能不存在唯一解，导致算法在两种可能解之间来回迭代而无法收敛。
- 可解释性问题：也许最终的聚类结果很难给出合理的解释。这通常是 *k* 均值聚类最棘手的问题。

尽管有很多问题，但是 *k* 均值聚类算法因为其逻辑简单，运行速度非常快，因此在实际中得到了广泛的应用：包括市场营销、计算机视觉（图像分割）等领域都有较多的应用。*k* 均值聚类还可以与其他监督性学习算法结合起来使用，作为某些模型的起始估计值。

在 R 中，*k* 均值聚类同样可以用一行代码搞定：

```
kmeans(x, centers, iter.max = 10, nstart = 1,
       algorithm = c("Hartigan-Wong", "Lloyd", "Forgy",
                     "MacQueen"))
```

代码中的 `x` 是数据矩阵，矩阵中的每一列代表一个特征变量。最大的迭代次数由 `iter.max` 所确定，这里我们设定为 10，当然你也可以改为自己认为合理的数值。有很多具体的算法可以实现 *k* 均值聚类，你可以指定使用哪一个算法。

关于 *k* 均值聚类的背景知识

我们刚才不是已经详细地解释了这个算法的步骤了吗？它看起来十分简单嘛。其实不然，*k* 均值聚类其实经过了很长时间的发展。

1957 年 Hugo Steingaus 和 Stuart Lloy 各自独立地开发了这个算法的早起标准版本，当然还不叫 *k* 均值聚类。来自贝尔实验室的 James MacQueen 在 1967 年首次使用了这一名称。但直到 1982 年 James 关于该名称的论文才得到公开发表（之前是作为贝尔实验室的内部研究而没有对外发表）。

在 20 世纪 60 年代和 70 年代，*k* 均值聚类算法得到了很多改进，包括 Hartigan-Wong、Lloyd 和 Forgy 等学者都有所贡献。我们之前描述的算法步骤就来自于 Hartigan-Wong 开发的版本。

直到最近，还会有关于 *k* 均值聚类算法的新研究发表。其中值得提及的是 David Arthur 和 Sergei Vassilvitskii（现任职于谷歌）于 2007 年发表的 *k* 均值 ++ 算法。该算法通过最优化初始随机种子有效地避免了算法不收敛的问题。

3.3 练习：机器学习算法基础

继续我们上一章关于纽约市（曼哈顿）房地产的数据分析。相关信息可见 http://abt.cm/1g3A12P。

- 应用线性回归模型，分析房地产销售额。选择你觉得可能有用的预测变量，并且验证为什么线性回归模型适用于该数据。
- 可视化线性回归模型的参数和拟合情况。
- 用 k 近邻模做预测。确保你在训练模型之前已经抽离了一部分数据用作独立测试。找到相关的预测变量以及最优的 k 值以达到最小的预测误差。
- 将你的模型结论通过可视化和报告的形式展示给大家。
- 基于模型的分析结果，提出一些可行的建议。

答案

上一章我们讲述了如何清理和探索该数据集。如果你之前没有动手清理过该数据，现在真的是需要动手的时候了，因为在应用回归模型分析数据之前，数据必须是干净的。接下来的两段 R 代码，第一段是有关于如何针对该数据构建一个线性回归模型的，第二段是有关如何清理和准备数据以应用 k 近邻模型进行预测的。

示例R代码：房地产数据的线性回归模型

```
作者：Ben Reddy

model1 <- lm(log(sale.price.n) ~ log(gross.sqft),data=bk.homes)
## 想一想这里的代码是用来做什么的

bk.homes[which(bk.homes$gross.sqft==0),]

bk.homes <- bk.homes[which(bk.homes$gross.sqft>0 &
            bk.homes$land.sqft>0),]
model1 <- lm(log(sale.price.n) ~ log(gross.sqft),data=bk.homes)
summary(model1)

plot(log(bk.homes$gross.sqft),log(bk.homes$sale.price.n))
abline(model1,col="red",lwd=2)
plot(resid(model1))

model2 <- lm(log(sale.price.n) ~ log(gross.sqft) +
  log(land.sqft) + factor(neighborhood),data=bk.homes)
summary(model2)
plot(resid(model2))

## 为了更好的可解释模型，这里去掉模型中的截距项
model2a <- lm(log(sale.price.n) ~ 0 + log(gross.sqft) +
  log(land.sqft) + factor(neighborhood),data=bk.homes)
summary(model2a)
plot(resid(model2a))
```

```
## 加入 building type（建筑种类）变量
model3 <- lm(log(sale.price.n) ~ log(gross.sqft) +
  log(land.sqft) + factor(neighborhood) +
  factor(building.class.category),data=bk.homes)
summary(model3)
plot(resid(model3))

## 加入 neighborhood（邻居个数）和 building type（建筑）之间的交叉效应变量
model4 <- lm(log(sale.price.n) ~ log(gross.sqft) +
  log(land.sqft) + factor(neighborhood)*
  factor(building.class.category),data=bk.homes)
summary(model4)
plot(resid(model4))
```

示例R代码：房地产数据的k近邻模型

```
作者：Ben Reddy
require(gdata)
require(geoPlot)

require(class)

setwd("~/Documents/Teaching/Stat 4242 Fall 2012/Homework 2")

mt <- read.xls("rollingsales_manhattan.xls",
  pattern="BOROUGH",stringsAsFactors=FALSE)
head(mt)
summary(mt)

names(mt) <- tolower(names(mt))

mt$sale.price.n <- as.numeric(gsub("[^[:digit:]]","",
                              mt$sale.price))
sum(is.na(mt$sale.price.n))
sum(mt$sale.price.n==0)

names(mt) <- tolower(names(mt))

## 利用正则表达式清理和格式化数据
mt$gross.sqft <- as.numeric(gsub("[^[:digit:]]","",
                            mt$gross.square.feet))
mt$land.sqft <- as.numeric(gsub("[^[:digit:]]","",
                           mt$land.square.feet))

mt$sale.date <- as.Date(mt$sale.date)
mt$year.built <- as.numeric(as.character(mt$year.built))
mt$zip.code <- as.character(mt$zip.code)

## 使数据集更加标准化（设定房屋建筑的起始年份为 0、土地面积为平方英尺、售价为 0 的
或者负值的去掉等操作）
min_price <- 10000
mt <- mt[which(mt$sale.price.n>=min_price),]

n_obs <- dim(mt)[1]
```

```
mt$address.noapt <- gsub("[,][[:print:]]*","",
                    gsub("[ ]+"," ",trim(mt$address)))

mt_add <- unique(data.frame(mt$address.noapt,mt$zip.code,
                 stringsAsFactors=FALSE))
names(mt_add) <- c("address.noapt","zip.code")
mt_add <- mt_add[order(mt_add$address.noapt),]

# 找重复值并删除，这些重复值往往是一个地址，但邮编却不同
dup <- duplicated(mt_add$address.noapt)
# 删除这些重复值
dup_add <- mt_add[mt_add$dup,1]
mt_add <- mt_add[(mt_add$address.noapt != dup_add[1] &
          mt_add$address.noapt != dup_add[2]),]
n_add <- dim(mt_add)[1]

# 随机选取 500 个地址，不然程序永远跑不完
n_sample <- 500
add_sample <- mt_add[sample.int(n_add,size=n_sample),]

# 首先用手头的一个地址尝试一下
query_list <- addrListLookup(data.frame(1:n_sample,
  add_sample$address.noapt,rep("NEW YORK",times=n_sample),
  rep("NY",times=n_sample),add_sample$zip.code,
  rep("US",times=n_sample)))[,1:4]

query_list$matched <- (query_list$latitude != 0)

unmatched_inds <- which(!query_list$matched)
unmatched <- length(unmatched_inds)

# 尝试把 EAST/WEST 改作 E/W
query_list[unmatched_inds,1:4] <- addrListLookup
  (data.frame(1:unmatched,gsub(" WEST "," W ",
  gsub(" EAST "," E ",add_sample[unmatched_inds,1])),
  rep("NEW YORK",times=unmatched), rep("NY",times=unmatched),
      add_sample[unmatched_inds,2],rep("US",times=unmatched)))[,1:4]

query_list$matched <- (query_list$latitude != 0)
unmatched_inds <- which(!query_list$matched)
unmatched <- length(unmatched_inds)

# 尝试把 STREET/AVENUE 改作 ST/AVE
query_list[unmatched_inds,1:4] <- addrListLookup
  (data.frame(1:unmatched,gsub(" WEST "," W ",
  gsub(" EAST "," E ",gsub(" STREET"," ST",
  gsub(" AVENUE"," AVE",add_sample[unmatched_inds,1])))),
  rep("NEW YORK",times=unmatched), rep("NY",times=unmatched),
      add_sample[unmatched_inds,2],rep("US",times=unmatched)))[,1:4]

query_list$matched <- (query_list$latitude != 0)
unmatched_inds <- which(!query_list$matched)
unmatched <- length(unmatched_inds)
```

```
## 现在我们已经做得差不多了
add_sample <- cbind(add_sample,query_list$latitude,
  query_list$longitude)
names(add_sample)[3:4] <- c("latitude","longitude")

add_sample <- add_sample[add_sample$latitude!=0,]

add_use <- merge(mt,add_sample)

add_use <- add_use[!is.na(add_use$latitude),]

# 地理坐标值
map_coords <- add_use[,c(2,4,26,27)]
table(map_coords$neighborhood)
map_coords$neighborhood <- as.factor(map_coords$neighborhood)

geoPlot(map_coords,zoom=12,color=map_coords$neighborhood)


## - 使用 knn 函数
## - 当然可以用的方法还有很多，此处不讨论过多

map_coords$class <- as.numeric(map_coords$neighborhood)
n_cases <- dim(map_coords)[1]
split <- 0.8

train_inds <- sample.int(n_cases,floor(split*n_cases))
test_inds <- (1:n_cases)[-train_inds]

k_max <- 10
knn_pred <- matrix(NA,ncol=k_max,nrow=length(test_inds))
knn_test_error <- rep(NA,times=k_max)

for (i in 1:k_max) {
    knn_pred[,i] <- knn(map_coords[train_inds,3:4],
  map_coords[test_inds,3:4],cl=map_coords[train_inds,5],k=i)
    knn_test_error[i] <- sum(knn_pred[,i]!=
      map_coords[test_inds,5])/length(test_inds)
}

plot(1:k_max,knn_test_error)
```

大规模数据建模及算法

迄今为止我们分析过的数据都难称为大数据。如果遇到大型的数据，我们的模型和算法会发生怎样的改变呢？

在大多数情况下，大型数据都可以拆分为一系列小数据集之后再分别独立建模（设置可以使用同一个模型）。这通常叫作大数据的“碎片化”（sharding，碎片化就是指将大数据分割成很多小的部分并分配给不同的计算机进行建模，模型的参数估计会以分布的形式返回给建模者）。

碎片化固然是大规模数据建模的一个较为简单和直接的解决方案。然而，当数据的规模达到一定的级别，模型本身就需要优化以解决计算量过大的问题。比如，线性回归模型的参数估计过程严重依赖于矩阵的求逆操作。对于大数据来说，矩阵的求逆操作是不现实的，因此应该考虑如何近似化矩阵的求逆以减轻计算负担，从而使得线性回归可以应用于分析大数据。

基于大数据分析需求的模型优化是大数据领域的研究前沿，它需要技术和理论的同步发展。Edinburugh 大学的 Peter Richtarik 于 2013 年有过一段关于大数据下模型优化的演讲，他说道："在大数据领域，传统的基于最优化方法的模型，因为涉及大量的迭代操作，给计算造成了巨大的负担，有时候即便是一次迭代也很难在短时间内完成。因为大多数统计模型在开发之初都没有考虑过大数据的适用问题，就算在 10 年前我们也很难预见数据会达到如今的规模。因此，我们亟需一些简单的、数据友好的、内存要求低的的模型和算法，以适应大规模数据分析。解决平行计算框架问题（包括多核计算问题、GPU 计算问题以及计算机集成技术）才能有效解决大数据分析的海量计算问题。"

对该问题的讨论已经超出了本书的范畴，但是作为一名数据科学家，我们要知晓这些问题的存在。在第 14 章我们还会对其中的某些细节问题做进一步的讨论。

3.4 总结

本章介绍了机器学习领域三个基础算法，它们已经被广泛地应用于数据分析的各个领域。如果你已经理解和掌握了这三个算法，那么对于数据科学来说，你已经基本入门了。如果你还是觉得理解和掌握这些基本的算法有点困难，不用灰心丧气，这本来就不是一段轻松的学习旅程。

在很多情况下，对于预测和分类问题来说，线性回归模型都是最基础的模型。本章已经为大家展示了如何应用线性回归模型预测一个连续型的数值型变量，模型中可以使用多个预测变量。在第 5 章，我们还会更详细地讨论线性回归模型，我们还会学习到逻辑回归模型，它可以用来预测二元分类变量；第 6 章将讨论时间序列模型。在第 7 章，我们会为大家介绍有关模型中特征变量的选择问题。

k 近邻和 *k* 均值聚类是两种聚类算法，适用于把相似的事物归类的问题。对于聚类算法来说，最重要的就是距离的定义和模型的评价标准这两个问题。它们的解决方式都具有一定的灵活性和主观性。下一章我们会继续讨论有关聚类算法的问题，其中朴素贝叶斯算法将是下一章的重点。在第 10 章我们会聊到关于社交网络数据的分析问题，社交网络分析中的图聚类模型是非常有意思的研究领域。本书没有涉及的聚类算法有分层聚类和基于模型的聚类，有兴趣的读者可以自动找资料了解一下。

对于聚类分析感兴趣的读者，我们推荐 Hastie 和 Tibshirani 的著作 *Elements of Statistical*

Learning（《统计学习基础》，http://stanford.io/16hcTKn）一书，英文版由 Springer 出版社出版。想要深入理解贝叶斯视角下的回归模型，我们极力推荐 Andrew Gelman 和 Jennifer Hill 的著作 *Data Analysis using Regression and Multilevel/Hierarchical Models*（《基于回归和多层 / 分层模型的数据分析》）。

3.5　思维实验：关于统计学家的自动化

Rachel 在 2013 年 5 月参加了在伦敦帝国理工学院（Imperial College London）举办的大数据挖掘会议。来自于剑桥大学的 Zoubin Ghahramani 教授是会议的嘉宾之一，他在其中的一个研讨会上提到了“自动化统计学家”的概念，他说打造一个“自动化统计学家”是他计划要完成的长期项目之一。对于“自动化统计学家”这个概念你有什么想法呢？如果让你打造一个“自动化统计学家”你该如何着手呢？

第 4 章

垃圾邮件过滤器、朴素贝叶斯与数据清理

本章的贡献主要来自于 Jake Hofman 先生。Jake 不久前刚从雅虎研究院离职，随后加入了微软研究院。他曾在哥伦比亚大学主修物理学，并获得博士学位。毕业之后，他留任哥伦比亚大学并长期教授一门广受学生欢迎的数据建模课程。近期他也开始教授一门新课，课程名叫作“社会科学中的计算方法”。

让我们先来了解一下 Jake 的数据科学背景。他本人曾经给数据科学添加过一项必备技能，名曰“数据清理”。他说自己不知道为什么在数据清理上花了那么多的时间，然而，熟能生巧，他俨然已经成为了数据清理方面的权威专家。

4.1　思维实验：从实例中学习

首先看一下图 4-1，这看起来应该是某人电子邮箱首页的截图，图中每一行显示的是一封邮件的概要。

再仔细看一下，你可能已经发现某些邮件乍一看就能判断出是垃圾邮件。

但是现实的问题是，我们如何才能确认一封邮件是否是垃圾邮件呢？我们的大脑可以综合各方面信息对一封邮件的性质做出逻辑推断，那么你能否通过编程来模拟人脑识别垃圾邮件的过程呢？

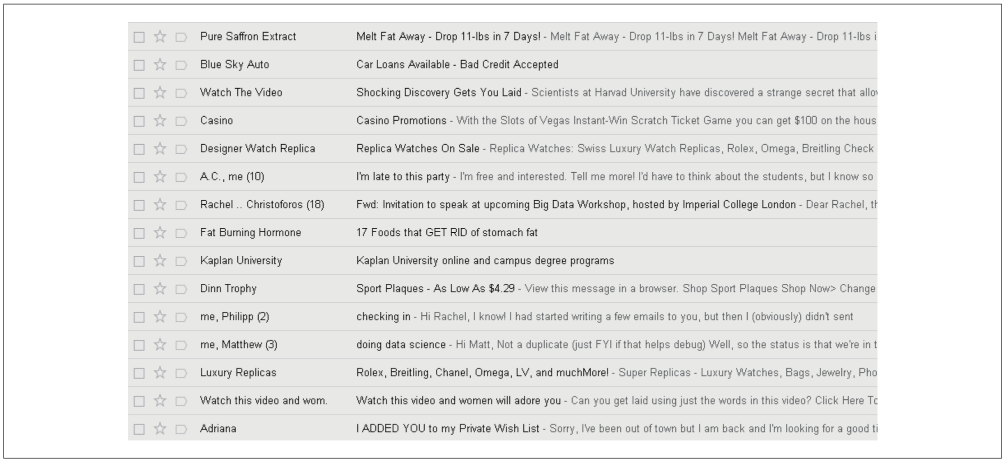


图 4-1：邮箱中的垃圾邮件

Rachel 的课程给了我们一些启示：垃圾邮件通会与某些特征指标紧密相关。

- 如果邮件中包含任何有关“伟哥”的信息，那么基本上可以确定是垃圾邮件。这是一条非常明晰的规则，但也许垃圾邮件的发送者也同样了解该规则，并可以通过改变拼写等方法成功地规避检查（其实，一些垃圾邮件制造者聪颖过人，只是他们没有把智慧用在该用的地方）。
- 邮件主题的字符长度，感叹号（或者其他标点符号）的使用频率等指标也可以作为垃圾邮件的特征指标。但也有例外：譬如雅虎公司的商标名是“Yahoo!”，其中的感叹号是该商标的设计元素，因此不能被一视同仁地视作垃圾邮件的标识。也就是说，垃圾邮件的特征指标设计也不能过于简单和严苛。

如果你想通过编程实现垃圾邮件的过滤，下面几点是我们的建议。

- 尝试使用概率模型。换句话说，避免使用过于简单的过滤指标。更好的办法是通过指标组合的方式，利用概率模型计算邮件是垃圾邮件的概率。在我们看来，概率模型非常适合用来搭建垃圾邮件过滤器。
- 在之前的章节中我们已经学习了 k 近邻以及线性回归模型，那么这两个模型可否用在此处呢？（答案是：不可以。大家可以先想一想为什么。）

本章我们将详细讨论如何应用朴素贝叶斯模型搭建一个垃圾邮件过滤器。 理论上看来，朴素贝叶斯是一个介于 k 近邻和线性回归模型之间的方法。至于原因，此处说来话长，让我们先从线性回归说起。

4.1.1　线性回归为何不适用

线性回归大家都不陌生了，它是我们工具箱中的必备神器。首先我们讨论一下该模型的适

用条件以及它为何不适用于垃圾邮件过滤问题。为了便于建模，我们将邮件数据先转换成数据矩阵，该矩阵的每一行代表某封邮件的信息（可以用 `email_id` 来标明邮件号），邮件中每个出现过的单词被称作“特征”，它们被置于矩阵的列上。举例来说，单词“伟哥”可以构成了矩阵的某一列，如果某封邮件出现了至少一次“伟哥”，那么在该邮件行的“伟哥”这一列上填上数字 1，代表“伟哥”在该封邮件中至少出现了一次；否则填上数字 0。当然，我们也可以在这个位置填上“伟哥”出现的实际次数。至于到底怎么填，要由数据分析人员事先确定。

回想一下前一章有关线性回归的内容，在建模之前我们首先需要一个关于邮件详情的训练数据集。在这个数据集中，每一封邮件已经被标明了是否是垃圾邮件。那么一个自然的问题就是，如何确定一封邮件是否是垃圾邮件呢？一种做法是人工核查每封邮件，并为每一封邮件打上是否为垃圾邮件的标签信息。该方法固然可行，但会耗费大量的人力物力财力。另一种做法是应用市面上已有的垃圾邮件过滤器为每一封邮件打上标签信息，比如使用 Gmail 声名远扬的垃圾邮件过滤系统。（此处一个自然的问题是，如果已经有了像 Gmail 这样一个业已成熟的垃圾邮件处理器，为啥我们还非要自己动手做一个呢？这是个好问题，不过在这里我们只是想通过演示如何搭建一个类似 Gmail 的过滤器去学习相关的算法）。一旦搭建好了模型，我们就可以用它预测一封没有标签的新邮件到底是否为垃圾邮件。[1]

不适用线性回归的首要原因是此处的目标 Y 变量（邮件是否为垃圾邮件）是一个二元变量（0 代表垃圾邮件，1 代表正常邮件）。我们知道，线性回归的预测值是一个连续性的变量，它可能是实数域上的任何一个数值，因此它的预测值不可能只是一个二元值。所以从模型输出值的性质来说，线性回归不适用于二元变量的建模，而更适用于连续性变量。

基本上可以说，线性回归对于垃圾邮件的识别问题是束手无策的，我们应该想办法找到一个更适用于二元变量的模型。但其实，如果我们生搬硬套，仍然尝试在 R 中使用线性回归模型搭建邮件过滤器，从理论上来说我们还是会得到回归模型的所有典型的输出结果。因为 R 软件本身不会告诉我们某个模型是否合适。我们还是可以用线性回归，用 R 估计出模型参数以及预测值。因为此时的预测值是一个连续性的数值，为了得到二元预测值（0/1）的输出结果，我们需要设定一个阈值，并且规定：如果预测值在阈值之上就设定预测值为 1，否则为 0。

即便如此生搬硬套，对于这样一个数据集来说，线性回归模型仍然会失败。为什么呢？因为对于邮件数据来说，变量的个数相比样本量的个数实在太大了。对于一个典型的邮件数据集，样本量的单位级在万左右（假设有 1 万封邮件），而其中所有可能的单词个数可能多达 10 万个，也就是说，变量个数为 10 万。传统的线性回归不能处理这种变量个数多于样本量的情形。线性回归模型参数估计的理论告诉我们，这会导致线性回归最小二乘解中

注 1：此处的意思是，训练数据集必须是已经做好标签的邮件数据，这样模型也有 Y 变量。

的矩阵不可逆。就算换个角度，从计算机存储的角度来说，处理这样规模的数据集，普通的个人计算机还是会显得捉襟见肘。

接下来很自然会想到，也许我们可以从 10 万个单词出挑选出最经常出现的 1 万个单词，这样最起码我们可以得到可逆的估计矩阵了。但即便如此，模型的估计结果也会非常不稳定。为了彻底解决变量过多的问题，假设我们可以把单词的变量个数缩减到了 100 个（比如，邀请垃圾邮件领域的专家仔细挑选最有用的单词变量），那么我们就顺利解决了模型的估计问题了。真的如此吗？其实不然，即便我们折腾到现在，还是没有解决一个最初也是最根本的问题：线性回归根本不能预测输出二元变量值！

番外话：垃圾邮件过滤器之现状

过去的 5 年中，研究人员开始使用随机梯度的方法解决我们刚才提到的模型估计矩阵不可逆的问题。例如，随机梯度方法可以和逻辑回归模型结合用来预测二元目标变量，而且这种方法能够考虑到单词变量之间的相关性。然而，在垃圾邮件过滤问题上，朴素贝叶斯模型显示了强大的适用性，它足够简单，效果也足够好。

4.1.2 *k* 近邻效果如何

听起来朴素贝叶斯模型十分诱人。别着急，我们很快就会讲到它。在这之前不妨让我们先回顾一下之前学习过的 *k* 近邻模型。对于任何模型来说，首要的任务还是要选择一些特征变量。沿用之前的做法，我们仍然使用单词作为特征变量。如果一个单词出现在邮件中，相应的变量值为 1，否则为 0。对于 *k* 近邻模型来说，最重要的莫过于确定“近邻”的含义。在什么样的情况下，两封邮件才可谓“近邻”呢？或者说，两个给定特征变量的邮件到底有多“近邻”呢？一个最直接的想法就是，当这两封邮件包含的单词十分相似时可将它们视为“近邻”。

之前的变量个数大于样本量的问题在这里同样困扰着我们，虽然在 *k* 近邻方法中我们不会遇到计算逆矩阵的问题，但是我们还是有一个大麻烦：数据空间的维度太高了。我们有 1 万封邮件和 10 万个单词，这就代表数据的空间高达 10 万维。从计算量上来说，在如此高维度的空间里计算相似度是很难的。

但其实计算量也不算是个大问题，最主要的问题是，在如此高维度的空间内，即便是两个最近的“邻居”也会相距深远。这通常称作“高维诅咒”，它会严重影响 *k* 近邻的模型效果。

番外话：数字识别

图 4-2 是 10 行从 0~9 的手写数字，假设我们想用一种算法识别这些手写数字，那么 k 近邻会是个不错的选择。

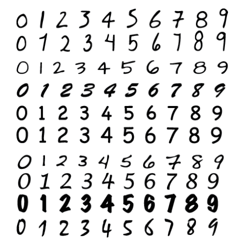


图 4-2：手写数字

要想建模，首先需要处理手写图像数据，将其转换成可以分析和建模的形式。譬如说，我们可以用一个 16×16 的像素框表示一个手写数字，框内的每个像素的亮度值代表了这个数字在该像素上的表示强度。然后再将这个 16×16 的像素框拉直成一个长度为 256 的向量。这样做的目的是便于我们接下来应用阿基米德测度计算距离。换句话说，我们成功地将每一个手写数字转化成了一个长度为 256 的向量。这样一来，两个手写数字的差异程度则可以用两个向量的阿基米德距离来表示。向量的阿基米德距离就是向量所有元素的“离差平方和”。在空间上解释就是两个向量间的点对点长度。完成上述的形式转化和距离定义之后，我们就可以着手建模了。

k 近邻的模型效果会随着近邻个数的改变而改变。我们需要调整好 k 的大小以尽量避免类似过拟合的现象出现。如果你已经认真做完了形式转化和距离定义的工作，并仔细地挑选了一个合理的 k 值，那么 k 近邻的模型效果通常都不会让你太失望。即使是将算法扩展到一个较大的数据集，模型准确度也通常会达到 97%。

模型最后的预测效果通常用一个“混淆矩阵”表示。假设模型的任务是将识别结果分到可能的 k 个类别中（在这个手写数字识别的例子中，$k=10$），那么混淆矩阵就是一个 $k\times k$ 的矩阵。该矩阵中的列代表实际的标签值，行代表了模型预测的标签值。因此该矩阵的第 (i, j) 个元素就代表有多少个结果，其实际的标签是 i 却被模型预测成了标签 j。给定一个混淆矩阵就可以轻松地得到模型的预测准确度：也就是所有被正确预测的标签数占总标签数的比例。在前一章中，我们介绍了误分率的概念，而这里的预测准确度 = 1– 误分率。

4.2 朴素贝叶斯模型

我们似乎走投无路了，因为线性回归和 k 近邻两大神器在垃圾邮件过滤问题上似乎都不太给力。不要灰心！接下来我们就立马介绍朴素贝叶斯模型，对于垃圾邮件过滤问题，它可以快刀斩乱麻。

4.2.1 贝叶斯法则

让我们通过一个更为简单的例子感受一下为什么朴素贝叶斯会大有作为。设想我们在检测一种罕见的疾病，其在人群中的发病率只有 1%。我们的测试方法非常适用于此病，具有很高的精度，但也并非 100% 的完美。其表现为：

- 99% 的病人可以被检测出患有此病（测试结果为阳性）[2]；
- 99% 的健康人可以被检测出未患此病（测试结果为阴性）[3]。

那么现在的问题是，如果一个人的测试结果为阳性，那么他真正患有此病的概率是多少呢？

让我们先用一个笨方法回答这个问题。假设有 100 × 100 = 10 000 个群众参于了测试，那么根据发病率，这个人群中会有 100 个人患有此病，而剩下的 9900 个人是健康的。该测试的结果是 100 个患病人口中会有 99 个被检测为患有此病；然而同样的，9900 个健康人口中同样也会有 99 个会被检测为患有此病。也就是说，如果一个人的测试结果为阳性，那么他真实的患病概率是对半开，50%！ 图 4-3 中的树形逻辑图更好的刻画了我们刚才的推理过程。

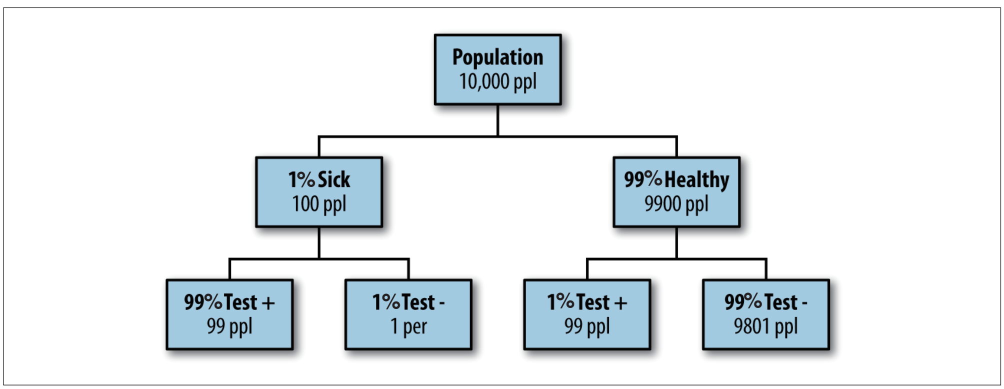


图 4-3：树形逻辑推断图

我们不如把刚才的推理过程用公式表示一下，这样会显得我们比较聪明。回想一下你可能上过的初等统计学的课程，给定两个事件 x 和 y，其各自发生的概率分别为 $p(x)$ 和 $p(y)$。它们联合发生的概率（表示为 $p(x, y)$）以及它们相互发生的条件概率（比如说 $p(y \mid x)$ 就表示给定事件 x 发生的情况下，事件 y 发生的概率）有如下关系：

$$p(y \mid x)p(x) = p(x, y) = p(x \mid y)p(y)$$

注 2：这也叫作“真阳性”。

注 3：这也叫作“真阴性”。

应用此式，我们可以得到贝叶斯法则并进而得到关于 $p\ (\ y\ |\ x)$ 的概率表示（此处假设 $p(x) \neq 0$）：

$$p(y \mid x) = \frac{p(x \mid y)p(y)}{p(x)}$$

其中的分母项 $p(x)$ 可视作一个“正则常数”，并且通常来说得到它的值相对比较容易。结合之前的例子，这里 y 对应于事件“患有此病”，此处用 Sick 表示；x 对应于“测试结果为阳性”，此处用 + 表示。应用上述的贝叶斯公式就可以计算出在给定测试结果为阳性的条件下某人真实患病的概率：

$$p(sick \mid +) = \frac{p(+ \mid sick)p(sick)}{p(+)} = \frac{0.99 \cdot 0.01}{0.99 \cdot 0.01 + 0.01 \cdot 0.99} = 0.50 = 50\%$$

结果与我们使用笨方法得出的结论完全一致！

4.2.2　个别单词的过滤器

准备工作已经做得比较到位了。我们应该捋起袖子，想一想到底如何应用如此简单的贝叶斯法则搭建一个靠谱的垃圾邮件过滤器呢？的确，如果一封邮件中有“伟哥”这样的单词，那么它极可能就是一封垃圾邮件。但是这压根还没完啊，我们还需要考虑邮件中可能出现的形形色色的各种单词呢！

不如让我们把每个单词抽象出来，各个击毙。这里，单词用“word”表示，垃圾邮件用“spam”表示。应用贝叶斯公式我们可以计算出，如果一个单词出现，该邮件可能是垃圾邮件的概率：

$$p(spam \mid word) = \frac{p(word \mid spam)p(spam)}{p(word)}$$

如果有足够多的已经做好标签的训练数据，这个等式右边的诸项都很容易计算出来。具体来说，如果用“ham”表示正常邮件，那么我们只需计算几个概率值：$p(word|spam)$、$p(word|ham)$、$p(spam)$ 以及 $p(ham) = 1 - p(spam)$。等式右项中的分母部分我们已经知道怎么算了（如果不会，看下之前医学检测的例子）：

$$p(word) = p(word \mid spam)p(spam) + p(word \mid ham)p(ham)$$

可以说我们已经把问题简化成了一个计数的问题：通过计算所有邮件中垃圾邮件的比例就可以得到概率 $p(spam)$，然后在所有的垃圾邮件中计算某一个特定单词出现的频率，我们又可以得到概率 $p(word|spam)$。更进一步，在所有的正常邮件中计算一个特定单词出现的频率我们又轻松的得到了概率 $p(word|ham)$。

大功告成！贝叶斯公式加上简单的计数工作解决了所有的问题，接下来就是你自己动手的

时候了。上网把 Enron 公司的电子邮件数据下载到你的个人电脑上（https://www.cs.cmu.edu/~enron/），接着就开始动手在这个数据上搭建一个垃圾邮件过滤器吧！这将意味着我们会替 Enron 公司的雇员们搭建一个全新的垃圾过滤系统，这个系统比他们正在使用的现存过滤系统要有效得多。当然，我们会根据 Enron 公司对于垃圾邮件的定义来设定关键词，使得新的过滤系统更适合该公司的需求。（这也意味着，如果从 2001 年开始，垃圾邮件的制造者们也从这些数据中学到了一些规则，他们就会在发送垃圾邮件的时候尽力规避，那么我们搭建的过滤器的功效可能就要大打折扣了。）

我们可以用 bash 写一段外壳脚本实现这个功能，记得 Jake 之前就是这么干的。该段代码的功能是从网上下载垃圾邮件数据集并解压缩到一个新的文件夹中。每一封邮件都对应一个文本文档，并且垃圾邮件和正常邮件应该置于不同的文件夹中以便区分。

这个邮件数据库中的某些统计量指标似乎唾手可得。比如，我们可以数出来总共有 1500 封垃圾邮件和 3672 封正常邮件，因此 $p(spam)$ 和 $p(ham)$ 的概率值已经在我们手中了。使用命令行工具，我们可以进一步在垃圾邮件文件中计算出单词“meeting”出现的次数：

```
grep -il meeting enron1/spam/*.txt | wc -l
```

我们得到的数值是 16。在正常邮件文件夹中重复上述操作，得到了数值 153。有了这些信息便可以轻易地计算出给定一封邮件中出现过单词“meeting”，它可能是垃圾邮件的概率：

$$\hat{p}(spam) = 1500/(1500 + 3672) = 0.29$$

$$\hat{p}(ham) = 1 - 0.29 = 0.71$$

$$\hat{p}(meeting \mid spam) = 16/1500 = 0.0106$$

$$\hat{p}(meeting \mid ham) = 153/3672 = 0.0416$$

$$\hat{p}(spam \mid meeting) = \frac{\hat{p}(meeting \mid spam) \cdot \hat{p}(spam)}{\hat{p}(meeting)} = \frac{0.0106 \cdot 0.29}{(0.0106 \cdot 0.29 + 0.0416 \cdot 0.71)} = 9\%$$

很明显我们根本不需要一个复杂的程序去实现上述简单的计算操作。

接下来让我们尝试一些别的单词，应用贝叶斯法则可以计算出如果一封邮件中出现了个单词，该邮件可能是垃圾邮件的概率。下面是一些计算结果，看起来似乎效果不错。

- money（金钱）：80%
- vigra（伟哥）：100%
- enron（安然公司名）：0%

现在模型已经工作了，但其实细想想，因为我们使用了 Enron 公司关于垃圾邮件的定义，模型很容易过拟合。也就是说，对于该模型我们不能过分自信，它很可能只能用在 Enron 公司的数据上。比如说，我们能否拍胸脯说，如果一封邮件中主要出现“伟哥”这一单词就铁定是一封垃圾邮件呢？肯定不能，因为从常理上来说，我们很容易写一封正常的邮件

并在里面开玩笑的写上“伟哥”这一单词。同样的道理，我也可以编造一封垃圾邮件，但只要在其中添加“enron”一词，它便可以轻松通过我们刚刚搭建的过滤器的检查了。

4.2.3　直通朴素贝叶斯

现在让我们把所有单词的信息都利用起来，搭建一个真正的朴素贝叶斯模型。每封邮件都可以表示为一个二元向量，这个向量的第 j 个元素是 0 还是 1 取决于第 j 个单词是否出现在这封邮件中（出现为 1，否则为 0）。向量的长度取决于总共要考虑的单词个数。如果要考虑所有在邮件中出现过的单词，那么这个向量必然会很长。

此时模型的输出值就是在给定一份邮件的标签值之后（也即知道它是否是垃圾邮件之后），这封邮件所代表的向量中的单词一起出现的概率。用 x 表示一封邮件的单词向量，x_j 表示向量中的某个元素，下标 j 代表某个单词在向量中的位置。c 代表垃圾邮件，那么上句中的概率值为：

$$p(x \mid c) = \prod_j \theta_{jc}^{xj}(1-\theta_{jc})^{(1-xj)}$$

式中的 θ 代表某个单词在垃圾邮件中出现的概率。上一节我们已经阐述了如何用计数的方法得到这个概率，此处我们假设每个单词在垃圾邮件中出现的概率都已经计算出来了，因此视为一个已知量。

朴素贝叶斯模型的核心概念是*独立性*，这也是为什么我们能够在上式的右边使用连乘符号的原因：因为我们假设单词之间出现与否是相互独立的。这个独立性假设也正是朴素贝叶斯方法中“朴素”一词的含义。这个假设在现实中是很难成立的，比如某些单词会倾向于成双结对的出现。因此在做独立性假设的同时，我们便忽略了这种可能同时出现的情形。[4]

回到刚才的等式中，对于概率连乘我们通常会在等式两边取对数，将连乘转变成连加的形式：

$$\log(p(x \mid c)) = \sum_j x_j \log\left(\frac{\theta_j}{1-\theta_j}\right) + \sum_j \log(1-\theta_j)$$

> 在连乘项中如果存在很多接近 0 的数值很可能会导致数值计算的不稳定。取对数恰好也解决了这个问题。

上式中的 $\log\left(\frac{\theta_j}{1-\theta_j}\right)$ 与邮件本身没有关系，而只取决与某个单词本身的性质。我们可以把它重命名为 w_j。同样假设 $\sum_j \log(1-\theta_j) = w_0$，于是上式可简化为：

注 4：当然，独立性假设还排除了很多别的情形，并不仅限于“单词同时出现”的情况。

$$\log(p(x \mid c)) = \sum_j x_j w_j + w_0$$

式中唯一跟邮件本身有关的就是 x_j 了，这个也不难计算。

现在我们终于可以集合所有得到的信息计算 $p(x \mid c)$ 了，然后贝叶斯法则会帮助计算出我们真正想知道的概率值 $p(c \mid x)$：贝叶斯公式中的其他部分的计算，相比 $p(x \mid c)$ 都较为简单。如果你只关心一封邮件更有可能是垃圾邮件还是正常邮件，那么你甚至不需要计算贝叶斯公式中的其他部分，而只需要计算 $p(x \mid c)$。因为只有它是跟邮件本身有关的变动项。

注意到没，最后的等式跟线性回归模型的等式十分类似。唯一不同的地方在于对于参数 w_j 的估计我们在这里使用了贝叶斯公式而不是通过计算逆矩阵。

朴素贝叶斯模型的精度很高，并且非常“实惠”。如果已经准备好了一个已经做好标签的数据集，可以立即上马朴素贝叶斯模型，它的训练非常容易。即便是过滤成千上万封邮件，我们要做的也仅仅是在垃圾邮件和正常邮件中数一数单词们出现的频率。如果有更多的数据，模型的更新也非常得迅捷：只要更新相应单词的频率就可以了。在实际应用中，通常会有一个简单的基础模型以备使用，我们要做的就是在这个模型基础上加以改良并个性化地运用到我们自己手中的数据上。所以，即便市面上有很多花里胡哨的复杂模型，而我们不妨用用朴素贝叶斯这样简单有效的模型。

如果你想更深入地了解贝叶斯法则和朴素贝叶斯模型，以下是一些不错的参考资料：

- “Idiot’s Bayes - not so stupid after all”（“给笨蛋的贝叶斯统计学：其实没有那么笨”，整篇文章都在论述为什么贝叶斯是个好方法；参见 http://goo.gl/sD86Oj）；
- “Naive Bayes at Forty: The Independence Assumption in Information”（“不惑之年的朴素贝叶斯方法：信息中的独立性假设的价值”，参见 http://goo.gl/JqjRg3）；
- “Spam Filtering with Naive Bayes - Which Naive Bayes?”（“垃圾邮件过滤与朴素贝叶斯模型——什么是贝叶斯统计学”，参见 http://goo.gl/74CxlZ）。

4.3　拉普拉斯平滑法

还记得刚才的 θ_j 吗？它代表垃圾邮件中某个单词出现的概率。仔细想想，它无非就是一个商数：$\theta_j = n_{jc}/n_c$，其中 n_{jc} 代表在该垃圾邮件中该单词出现的总次数，而 n_c 则代表在所有邮件中（包括垃圾邮件和正常邮件）该单词出现的次数。

拉普拉斯平滑法乍看起来相当无厘头：

$$\theta_{jc} = (n_{jc} + \alpha)/(n_c + \beta)$$

只要满足 n_{jc} 是一个概率值（位于 0 和 1 之间），α 和 β 可以任意取值。比如，我们可以设定 α =1，β =10，但是其背后的道理却很难一眼就看出来。其实拉普拉斯平滑与刚才讲到的

朴素贝叶斯关系相当密切。如果在朴素贝叶斯模型中加入先验信息，并应用最大似然估计法估计参数，拉普拉斯法变得花哨起来了。用 ML 表示最大似然估计法，数据用 D 表示，那么参数的最大似然估计可以表示为：

$$\theta_{ML} = argmax\theta p(D \mid \theta)$$

而刚才我们提到的$\theta_j = n_{jc} / n_c$就是最大似然估计的结果，它的含义是：这样的估计最有可能生成这样的数据 D。朴素贝叶斯中最关键的假设条件就是“独立性假设”，如果该假设在这里仍然成立，那么对于每一个 j 我们都可以分别用一个θ_j最大化其相应的似然值：

$$\log(\theta_j^{n_{jc}}(1-\theta_j)^{n_c - n_{jc}})$$

对其取一阶导并令为 0，可以得到：

$$\hat{\theta}_j = n_{jc} / n_c$$

这正是我们在朴素贝叶斯模型中得到的结果。这意味着，如果“独立性假设”成立，朴素贝叶斯中θ的估计值恰好就是它的最大似然估计值。

如果加上先验信息，结合似然函数便可以得到后验估计值，进而得到最大后验似然估计（MAP）：

$$\theta_{MAP} = argmax\ p(\theta \mid D)$$

这相当于在回答这样一个问题：如果已经观测到了所有数据，哪个θ值是最有可能的?

贝叶斯法则在这里起到了关键性的桥梁作用，它把估计量 θ_{MAP} 与 $p(D \mid \theta) \cdot p(\theta)$ 紧密地联系了起来：它们是严格地正比例关系。其中，$p(\theta)$ 称作“先验信息”，通常我们要人为地赋予其一个合理的概率值。如果我们假设 θ 服从一个最简单地 Beta 分布 $\theta^{\alpha}(1-\theta)^{\beta}$，那么 θ 的最大后验估计量 θ_{MAP} 就是本节刚开始所展示的拉普拉斯平滑值。

假设的先验分布合理吗

θ 的含义是：如果邮件中出现某个单词那么该封邮件可能是垃圾邮件的概率。假设它服从一个 Beta 分布，那么只要 α 和 β 都大于 0，它的概率分布在 0 和 1 两侧的取值都接近于 0。我们说过，我们不能 100% 地确认一个单词的出现就代表这是一封垃圾邮件，因此这样的概率分布假设应该是合乎常理的：在垃圾邮件中，几乎没有单词会绝对出现 / 不出现，我们不想表现得过于极端。

然而，如果 α 和 β 取比较大的值，那么 θ 的大部分值都会集中在其分布的中间部分。也就是说大部分的 θ 都十分相近。从垃圾邮件过滤的角度来看，大部分的单词都以相似的概率出现在垃圾邮件和正常邮件中，这似乎不是一个合理的假设，因为总有一些单词与垃圾邮件的关系更加密切。

一个折中的方法是把 α 和 β 设定为一个很小的正值，比如说 0.2。这样我们的模型才既不会过于极端，也不会过于平庸。对于先验信息分布，规矩是，数据量愈大可以愈发放松对它的假设。因为，大样本量代表的似然函数部分的效果会极大地掩盖先验信息的影响，导致我们没有必要在先验信息的分布上大动干戈。但是如果数据量不大并且我们对先验信息可能的分布形态比较自信，那么我们可以设定一个较为主观的先验分布。

4.4 对比朴素贝叶斯和 *k* 近邻

在刚才的模型中，α 和 β 两个参数通常称作“超参数”或者“伪计数”。数据科学家对选择什么样的超参数值具有完全的自主权。它们看起来似乎花哨，但只要从后验分布的角度来分析其实也十分简单。相比较而言，k 近邻只有一个参数需要调整：近邻的个数 k。前面的讨论中，我们发现朴素贝叶斯的终极模型形态跟线性回归模型十分相似。没错，它确实是一个线性模型（在这里是一个线性分类器），但是 k 近邻却不是。k 近邻模型还会受到“高维诅咒”的挑战，而朴素贝叶斯却没有这个苦恼。k 近邻的方法不需要加以训练，拿来即用；而朴素贝叶斯模型却需要多加以训练以持续改善它的模型效果。然而，它们的共同之处在于：它们都属于监督性学习模型：在建模之前我们已经知道哪些是垃圾邮件而哪些是正常邮件。也就是说，我们事先已经知道了真理，而真理监督着我们把模型做好。

4.5 Bash代码示例

```
#!/bin/bash
#
# 文件名：enron_naive_bayes.sh
#
# 文件描述：训练一个简单只包含一个单词的朴素贝叶斯垃圾邮件过滤器
# 使用的是安然（Enron）公司提供的数据集
#
# usage: ./enron_naive_bayes.sh <word>
#
# 要求：
#   wget
#
# 作者：jake hofman (gmail: jhofman)
#

# 如何使用这段代码
if [ $# -eq 1 ]
    then
    word=$1
else
    echo "usage: enron_naive_bayes.sh <word>"
    exit
fi
```

```
# 如果文件不存在，则去网上下载得到
if ! [ -e enron1.tar.gz ]
    then
    wget 'http://www.aueb.gr/users/ion/data/
    enron-spam/preprocessed/enron1.tar.gz'
fi

# 如果文件夹本身不存在，解压缩刚才得到的 .tar.gz 文件
if ! [ -d enron1 ]
    then
    tar zxvf enron1.tar.gz
fi

# 切换到 enron1 目录
cd enron1

# 计算得到总垃圾邮件个数，总正常邮件个数以及总邮件个数
Nspam=`ls -l spam/*.txt | wc -l`
Nham=`ls -l ham/*.txt | wc -l`
Ntot=$Nspam+$Nham

echo $Nspam spam examples
echo $Nham ham examples

# 计算得到垃圾邮件和正常邮件中包含该单词的邮件个数
Nword_spam=`grep -il $word spam/*.txt | wc -l`
Nword_ham=`grep -il $word ham/*.txt | wc -l`

echo $Nword_spam "spam examples containing $word"
echo $Nword_ham "ham examples containing $word"

# 使用 bash 里的 bc 计算器计算相关概率值
Pspam=`echo "scale=4; $Nspam / ($Nspam+$Nham)" | bc`
Pham=`echo "scale=4; 1-$Pspam" | bc`
echo
echo "estimated P(spam) =" $Pspam
echo "estimated P(ham) =" $Pham

Pword_spam=`echo "scale=4; $Nword_spam / $Nspam" | bc`
Pword_ham=`echo "scale=4; $Nword_ham / $Nham" | bc`
echo "estimated P($word|spam) =" $Pword_spam
echo "estimated P($word|ham) =" $Pword_ham

Pspam_word=`echo "scale=4; $Pword_spam*$Pspam" | bc`
Pham_word=`echo "scale=4; $Pword_ham*$Pham" | bc`
Pword=`echo "scale=4; $Pspam_word+$Pham_word" | bc`
Pspam_word=`echo "scale=4; $Pspam_word / $Pword" | bc`
echo
echo "P(spam|$word) =" $Pspam_word

# 返回到之前的目录
cd ..
```

4.6 网页抓取：API和其他工具

作为数据科学家，数据并不总是现成的。你经常需要自己去想搜集数据的方法，如提出问题，并尝试研究和解决它。API 就是采集数据的工具之一。API 全称是“应用程序编程接口”，网站会通过 API 为开发者提供网站使用的数据。这些数据通常都具有固定的格式以便于后期处理。（API 的用处还有很多，数据接口只是其中一小部分。）一般来说，使用 API 之前需要注册并获取一个“密钥”。密钥其实就是一串长长的密码，用来识别你已经注册并申请使用 API。很多大型网站都会提供 API，其中比较著名的有《纽约时报》的官方网站（http://developer.nytimes.com/docs）。

> 关于使用 API 的警告：在使用之前要仔细阅读网站关于 API 使用的条款细则。另外，一些网站对 API 的使用设定了诸多限制，包括你能获取的数据类型以及免费账户的访问频率等。

API 数据的格式多种多样，业界还没有统一的 API 数据格式标准，因此从不同的网站得到的数据格式可能不尽相同。其中 JSON 是较为常见的一种。

幸运的是，我们可以使用雅虎的 YQL 语言（http://developer.yahoo.com/yql/）整合不同的 API 数据格式。你所要做的就是进入雅虎开发者网络，用类似下方的 SQL 语法抓取一些常见网站的 API 数据，YQL 会将它们整合输出成统一的格式。Python 可以识别 YQL 的输出格式并一次性读取所有的数据。

```
select * from flickr.photos.search where text="Cat"
and api_key="lksdjflskjdfsldkfj" limit 10
```

这是标准输出，你只需要在 Python 中解析一次。

如果你想抓取的网页没有提供 API 那该怎么办呢？

在 Firefox 火狐浏览器中有一款名叫 Firebug 的插件，它能够完整的扫描网页并提取网页 HTML 中你想要的信息。技术上来看，Firebug 就是把网页转换成了可供浏览和编辑的 HTML 文本。也就是说，HTML 类似于一张完整的网页地图，而 Firebug 就是你的导游。

当你在 HTML 文本内找到想要的内容之后，再用诸如 curl、wget、grep、awk、perl 等编程工具写一两行简短的代码，想要的东西就到手了。我们其实还可以用 Python 或者 R 把上述操作全部自动化。

另外还有一些不错的文本解析工具，此处我们提及几个。

- lynx 以及 lynx--dump（http://lynx.browser.org/）
  如果你比较怀旧，这一款应该非常适合你。它的界面还保持着 20 世纪 70 年代的风格。额……也许没有那么久远，但说是 1992 年的风格应该不为过。

- Beautiful Soup（http://www.crummy.com/software/BeautifulSoup/）
  非常稳健但速度很慢。
- Mechanize（http://mechanize.rubyforge.org/Mechanize.html 或 https://pypi.python.org/pypi/mechanize/）
  此款非常酷炫，但是不支持 JaveScript。
- PostScript（https://en.wikipedia.org/wiki/PostScript）
  适用于图像解析与分类。

思维实验：图像识别

如何训练计算机识别一张照片的内容是风景照还是人物肖像照呢?

首要的问题是，怎么才能弄到建模用的贴好标签的数据呢? 如已经有了这些照片，可以雇一些员工，人工审核这些照片并给它们贴上标签。这似乎不太现实。其实一个简单的方式是从 flickr 网站（http://www.flickr.com/）上抓取一些已经被用户贴好标签的照片。

取得数据之后，接下来要对照片进行数据预处理。每张照片都可以表示为一个 RGB（红绿蓝）密度直方图。也就是照片中的每一个像素点的色彩都可以表示为该点的三原色组合的密度值。红绿蓝是三个基本色（也成为三基色或者三原色），其中每个颜色的强度大小都可以用一个介于 0 和 255 之间的数值表示。而所有的颜色都可以表示为红绿蓝三基色的某个特定的强度组合，也就是一个 RGB 密度直方图。

一个图像可以有三个密度直方图，分别为 R、G 和 B；对应于红、绿、蓝三基色在该图像上的密度分布。因为 255 是个较大的数值，传统的直方图表示会变得较为杂乱。因此我们可以用块状直方图替代，这样我们需要表示的密度范围就缩减为 0~51 了。也就是说，一张图像可以表示为 15 个数值，因为我们总共需要表示 3 种基本颜色，而每种颜色又被细分成了 5 个色块。当然，我们在这里假设了每张照片的像素点个数是相同的。

最后，我们可以使用 k 近邻模型进行分类。比如说，我们可以挑选“蓝色”作为分类条件，k 近邻很可能会根据蓝色分布上的不同将风景照和人物肖像照成功分离开来。

4.7　Jake的练习题：文章分类问题中的朴素贝叶斯模型

该题的主要目的是把朴素贝叶斯模型应用到一个多标签文本的分类问题。首先，利用《纽约时报》官网的 API 从《纽约时报》网站的不同栏目中抓取一些近期的文章，再把每一篇文章转换成单词向量。我们的模型任务是：给定一篇文章的单词向量，预测这篇文章可能来自于《纽约时报》的哪一个栏目？现在让我们一步一步完成这项任务。

第一步，你需要去《纽约时报》官网注册，拿到“key”之后就可以获得下载文章的 API

权限了。事先仔细阅读一下 API 的使用须知，了解基本的 API 权限，再编写程序分别从“艺术”“商业”“讣告”“体育”和“国际”等 5 个栏目中下载 2000 篇最近一段时间发表的文章。（提示：下载文章时一定要注明文章到底来自哪个栏目板块，可以用 nytd_section_facet 这样的标签值以示区别。）你可以使用它们提供的控制台快速地了解一下该 API 的特性。编写代码的时候还需要注意，来自于不同栏目的文章需用单独的文件夹存储。格式方面，可以选择制表符定界格式，并在第一列上存储文章的 URL，第二列上存储本章的标题，第三列上存储文章的正文部分。

下载和整理完文章的数据之后，请编写程序实现一个最基本的朴素贝叶斯模型。所有的文章都可能出自 C 个栏目中某一个，用 y_i 表示第 i 篇文章的栏目标签，那么 $y_i \in 0, 1, 2, \cdots, C$。比如，第 0 类代表“艺术”，第 1 类代表“商业”……由此类推。每篇文章用一个稀疏的二元矩阵 $\boldsymbol{X}$ 表示，$\boldsymbol{X}_{ij}=1$ 则代表第 i 篇文章包含第 j 个单词。

先要通过计数估算出两个关键参数的取值：

$$\hat{\theta}_{jc} = \frac{n_{jc} + \alpha - 1}{n_c + \alpha + \beta - 2}$$

$$\hat{\theta}_c = \frac{n_c}{n}$$

其中 n_{jc} 代表 c 栏目中出现了单词 j 的文章数；n_c 代表 c 栏目中的总文章数，n 代表所有的文章数。α 和 β 前文已经有过介绍，它们是拉普拉斯平滑估计中的两个“超参数”。有了这些参数的估计值之后，就可以给文章 x 分类了。之前的讨论中，我们已经推导出了朴素贝叶斯的最终模型形态，它与线性回归非常相似。如果我们把栏目 0 看作基类，那么很容易计算出其他栏目类别相对此基类的对数发生比（log-odds）：

$$\log\left(\frac{p(y = c \mid x)}{p(y = 0 \mid x)}\right) = \sum_j \hat{w}_{jc} x_j + \hat{w}_{0c}$$

其中：

$$\hat{w}_{jc} = \log\frac{\hat{\theta}_{jc}(1 - \hat{\theta}_{j0})}{\hat{\theta}_{j0}(1 - \hat{\theta}_{jc})}$$

$$\hat{w}_{0c} = \sum_j \log\frac{1 - \hat{\theta}_{jc}}{1 - \hat{\theta}_{j0}} + \log\frac{\hat{\theta}_c}{\hat{\theta}_0}$$

对于每篇文章的分类，模型代码工作的流程大致是这样的：首先读取每篇文章的标题和正文，移除一些无关的字符（譬如标点符号），文本内容分词化并过滤掉一些停用词。该流程的主要目的是将文本中对于分类有用的词语解析并存储起来。在训练模型阶段，这些解析过的词语特征变量，联合我们主观设定的 α 和 β 值，便可以计算出分类模型中最关键的权重参数 $\hat{w}$。模型训练好之后，一篇待分类文章（当然每篇文章都需要经过上述的读取、移除、分词和过滤流程，以得到一样的词语特征变量）的词语特征变量会交给已经训练好的模型，模型会输出对应每个栏目类别的后验概率。

通常我们会将手中的数据随机分成两半，一半用来训练模型一半用来检测模型的预测效果：包括预测准确度和模型运行速度。因为 α 和 β 的设定较为直观，因此需要根据模型在测试集上的预测效果不断调试。对于一篇文章，模型会给出每个类别的后验概率值。所以我们会把具有最大概率值的类别分配给该文章。模型的分类效果可以用一个 5×5 的混淆矩阵表示。单词的分类能力以及文章的可分类型同样也非常有意思，比如说我们可以根据模型的预测效果列出“10 个分类能力最强的单词”，或者“10 篇最难分类的文章”等。

如果把刚刚的模型应用到其他数据集上效果会如何呢？比如说，应用到《纽约时报》早些时候的文章，或者是其他报纸的文章数据等？这称作模型的扩展性，你可以思考一下。

使用《纽约时报》的API： R代码示例

```
# 作者：Jared Lander
#
# 用硬编码写的 API 请求
theCall <- "http://api.nytimes.com/svc/search/v1/
article?format=json&query=nytd_section_facet:
[Sports]&fields=url,title,body&rank=newest&offset=0
&api-key=Your_Key_Here"

# 此处我们需要 rjson、plyr 和 RTextTools 软件包
require(plyr)
require(rjson)
require(RTextTools)

## 首先让我们看一个单独的 API 请求
res1 <- fromJSON(file=theCall)
# 结果多长
length(res1$results)
# 查看第一项
res1$results[[1]]
# 第一项的标题
res1$results[[1]]$title
# 得到的第一个结果被转换成了数据框（data.frame），并且可以在 R 的数据查看器中方便地
展示（使用 View）出来
View(as.data.frame(res1$results[[1]]))

# 把请求的结果转换成数据框的形式，该数据框应该是 10 行 3 列
resList1 <- ldply(res1$results, as.data.frame)
View(resList1)

## 接下来就可以应用到更多的请求了
# 创建一个字符串用来替换第一个 %s 和补偿第二个 %s
theCall <- "http://api.nytimes.com/svc/search/v1/
article?format=json&query=nytd_section_facet:
[%s]&fields=url,title,body&rank=newest&offset=%s
&api-key=Your_Key_Here"
# 生成一个空的列表（list）对象以存储 3 个请求的记过
resultsSports <- vector("list", 3)
## 在 0 到 2 的三个数值中循环迭代得到每一个 AIP 请求的值
for (i in 0:2)
{
```

```
    # 创建一个查询字符串，把第一个 %s 替换为 Sports
    # 把第二个 %s 替换为当前的 i 值
    tempCall <- sprintf(theCall, "Sports", i)
    # 应用该查询操作并得到相应的 json 返回值
    tempJson <- fromJSON(file=tempCall)
    # 把得到的 json 对象转换成一个 10x3 的数据框
    # 并保存为一个列表对象
    resultsSports[[i + 1]] <- ldply(tempJson$results,
    as.data.frame)
}
# 将该列表对象转换成数据框
resultsDFSports <- ldply(resultsSports)
# 创建一个新列，以表示该结果来自于 Sports 栏目
resultsDFSports$Section <- "Sports"

## 上述操作都是跟对 Sports 栏目的，也同样可以应用到 arts（艺术）栏目
## 此处的代码只做参考
resultsArts <- vector("list", 3)
for (i in 0:2)
{
    tempCall <- sprintf(theCall, "Arts", i)
    tempJson <- fromJSON(file=tempCall)
    resultsArts[[i + 1]] <- ldply(tempJson$results,
    as.data.frame)
}
resultsDFArts <- ldply(resultsArts)
resultsDFArts$Section <- "Arts"

# 将上述两个栏目的结果整合到一个数据框中
resultBig <- rbind(resultsDFArts, resultsDFSports)
dim(resultBig)
View(resultBig)

## 现在进行“标记化”（tokenizing）操作
# 创建一个英文的文献 – 检索词矩阵（document-term matrix），剔除其中的数字和停用词，提
取词干
doc_matrix <- create_matrix(resultBig$body, language="english",
removeNumbers=TRUE, removeStopwords=TRUE, stemWords=TRUE)
doc_matrix
View(as.matrix(doc_matrix))

# 此处分别创建训练和测试数据集
theOrder <- sample(60)
container <- create_container(matrix=doc_matrix,
labels=resultBig$Section, trainSize=theOrder[1:40],
testSize=theOrder[41:60], virgin=FALSE)
```

自然语言处理之历史背景

最近十分热门的自然语言处理（NLP）是计算机科学的一大研究前沿。刚刚讨论过的文本识别问题只是该领域研究中的冰山一角。NLP 可以解决的问题很多，包括机器翻译、语义分析、词性标注以及文本识别等。其中的机器翻译，通过算法将一种语言自动且实时的翻译成另一种语言。而本章讨论的垃圾邮件过滤问题恰好是一个文本识别问题。NLP 的研究甚至可以追溯到 20 世纪 50 年代。

第 5 章

逻辑回归

本章的贡献者是 Brian Dalessandro。Brian 是 Media6Degree（M6D）公司分管数据科学部门的副总裁，在学术界他也十分活跃。他还兼任 KDD 数据科学竞赛的联合主席。M6D 是一家从事在线广告业务的创业公司，其总部位于纽约。图 5-1 是 Brian 的数据科学知识构成，*y* 轴上的值被卡通化成了小丑（对应小值）和摇滚明星（对应大值）。

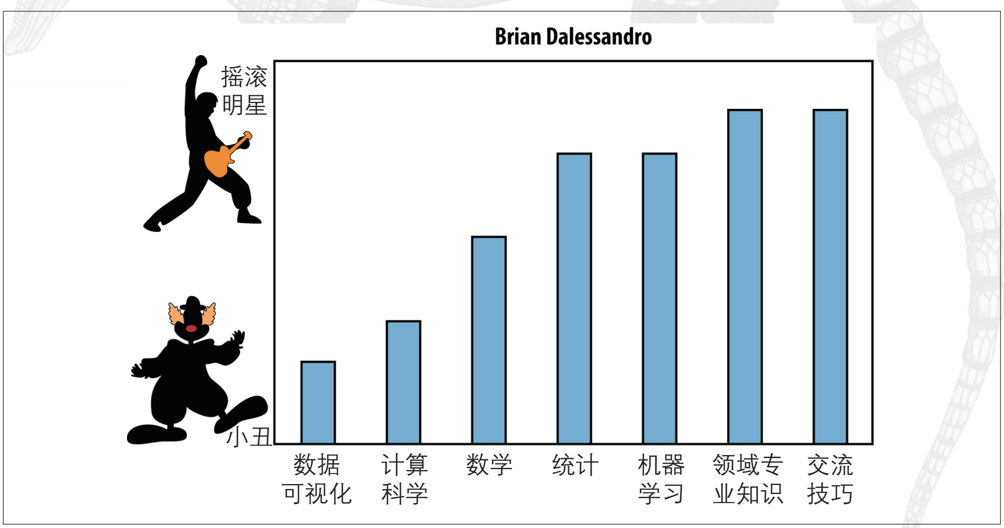


图 5-1：Brian 的数据科学知识构成图

Brian 曾来到我们的课堂为大家上了一课，主要内容是逻辑回归和评价。但在正式开始之前，他先做了两个思维实验。

5.1 思维实验

(1) 如果“大统一理论”真的成立，那么数据科学到底还有什么特别之处呢？假设“大统一理论”指的是对世界上万事万物运行规律的普适解释。这个问题引申出了一系列问题。

- 如果真有一个普适的理论，那么我们还需要研究数据科学这样的具体学科吗？
- “大统一理论”有没有存在的可能性？这个理论如果存在，那么它是否只存在于某一领域，比如说物理学？物理学是关于世界如何运转的学科，它强调精确性，比如说可以精确预见 100 年才出现一次的那颗彗星何时重返。
- 如果这个理论不可能存在，就说明物理学和数据科学是有本质区别的，那这种区别是什么？
- 两者的区别就只有准确度这一项吗？或者更广义地说，我们所能想到的东西，到底有多少能分别用这两种理论来解释？是不是因为我们在预测人类行为时，研究对象的行为会受到预测本身的影响，从而形成了一种反馈回路？

  若将科学看作一个统一的整体，可能对解答上述疑问会有所帮助。在这个统一体中，精确的物理学处于最右端，而越往左走就越混乱——研究者要面对更多的不确定性和随机性（也意味着更高的薪水）。那么诸如经济学、营销学和金融学这些学科又在科学体系中处于什么位置呢？

  如果数据科学像物理学一样，已经有一套业成熟的建模方式，那么要知道人们在何时会点击什么样的广告，就变得和预测火星探测器何时着陆一样容易。鉴于此，人们目前形成了普遍共识：无论是现在还是未来，我们都无法彻底了解这个世界。

(2) 数据科学值得称作“科学”吗？

不要低估了创意的力量——很多时候人们有了设想，却未能找到实现手段。而作为数据科学家，你应该有能力把设想转化为一个数学模型，这个模型在操作上会具备一些约束条件。你需要明确地知道问题所在，快速测度问题的方方面面，并且对它进行优化。而至关重要的一点是，在建模完成之后你要确保这个模型能够解决最初提出的问题。

数据科学中也是讲究艺术的，这主要体现在将人类实际问题和数学语言互为翻译转化的过程中。

经验告诉我们，这种转化问题的方式是没有标准答案的——可选的模型总是不止一种，相应的模型评价指标也有很多，甚至连最优化的方法都有很多选择。而数据科学之所以称作科学——给定原始数据、限制条件和问题描述——其恰恰在于这样的问题总是没有绝对普适的答案，我们需要经历一个迷宫一样的过程才能找到一个可能的最优解。每一种方案的选择都可以被视作一种假设，你需要具备利用精确的测试和实验方法来检验（验真或者证伪）这些假设的能力。

这样一种假设和检验的循环往复的过程给“数据科学”深深地烙上了“科学”的印记。具体来说，其“科学”的一面主要体现在下面三点。

- 如果你找到了一个最优的模型，坚持使用它！
- 如果你有一个新主意，把它与你之前的最优模型进行比较。通常，你需要思考一下如何设计好一对比较实验。
- 在能够 100% 确定之前，不停地实验（但也要尽量避免过拟合）。

5.2 分类器

本节的重点是如何选择一个好的分类器。分类器也就是分类模型：给定一些数据，它可以输出数据对应的某个类别或者是某些类别的概率值。上一章我们讲到的朴素贝叶斯模型以及 k 近邻都属于分类器的一种。表 5-1 中列出了一些需要分类器的常见情形。

表5-1：分类器的例子：想要回答的问题以及答案的形态

问　题	答　案
用户会点击这条广告吗	1 或者 0（会或者不会）
这个图像是哪个数字	0，1，2，…
这条新闻可能出自哪个栏目	“体育”“政论”等
这是一封垃圾邮件吗	1 或者 0
这个药治疗头疼有效吗	1 或者 0

让我们先讨论以下最简单的分类器：二元分类器（输出值为 0 或者 1）。

本章讨论的模型叫作逻辑回归，其他的…二元分类模型还包括决策树（第 7 章）、随机森林（第 7 章）、支持向量机以及神经网络等（本书没有涉及）。问题的背景是，给定一些数据和一个来自真实世界的分类问题，你需要决定：

(1) 使用哪一个分类器；
(2) 应用何种优化方法训练分类器；
(3) 选择什么样的损失函数；
(4) 哪些特征变量对建模有用；
(5) 如何评估模型的实际效果。

我们先讨论第一个问题：既然有这么多的分类器可以使用，那么我们到底应该选择哪一个呢？一个自然的想法是，每一个都尝试一下，然后从所有的备选模型中选出一个最好的。这样的笨办法是不可取的，因为在现实问题中总是有各种各样的限制条件，比如数据量有限，或者时间有限等。我们没有条件做大海捞针的工作。模型选择的问题往往不受重视，人们以为总是可以在众多模型中找到一个适用的模型，其实不然。下面我们就讨论一下现实问题中一些常见的限制条件。

5.2.1 运行时间

在 M6D 公司，我们每天要更新 500 个决策模型，因此我们格外重视模型的运行和更新速度。这里的速度，指的不仅仅是更新一个模型的速度，还包括模型真实的应用速度。后者我们通常称为模型的“运行时间”，它通常比前者要更加重要。

某些模型需要大量的“运行时间”，比如说 k 近邻模型：当模型数据空间很大时，预测一个新数据的类别需要计算这个数据点的 k 个“邻居”，因此需要把所有的新旧数据点都存储在内存中，这通常会耗费大量的“运行时间”。

而线性模型则不然，无论是模型更新还是用作实际预测，它的速度通常都令人满意。下一章你将看到，线性模型的更新过程只涉及新的数据，因此不需要把旧的数据也放在内存中，这极大地提高了模型的运行速度。一旦线性模型的参数估计完毕，只需要保存这些参数的估计值，预测新数据只涉及计算参数估计向量与新数据特征变量的点积的问题。

5.2.2 你自己

有一个限制条件常常被我们忽视，那就是我们自己！在应用模型分析数据的时候，要不时地拷问自己，是不是真的理解你正在使用的模型。

在这一点上，我们要诚实一点。如果不理解也没有关系，因为没有人能理解所有的模型，这也并不是成为一个数据科学家的必要条件。但是通常来说，如果你想发挥一个模型的最大效用，必须全面理解这个模型，了解模型参数的含义、模型的假设条件等方方面面。有时，你需要调整模型算法来拟合自己的数据。由于未能透彻理解手中的模型，人们经常不知不觉就陷入了过拟合的陷阱。

5.2.3 模型的可解释性

商用模型需要具备良好的可解释性，不然客户不会买账。有些模型天生可以说故事，比如决策树模型；而有些模型则是个黑匣子，你根本不知道里面发生了什么，比如随机森林模型。即便随机森林只是决策树模型的一个推广，在可解释性方面却有天壤之别。复杂的模型通常都很难解释，因此如果你没有足够的时间去解释一个复杂模型的结果，可以考虑牺牲一些准确度，把模型稍作简化。

一个典型的例子就是信用评分模型。法律规定，信用卡公司如果拒绝一个信用卡申请，必须给出充分的理由。 因此在信用评分模型中，决策树模型要比随机森林模型更加适用。法律没有规定所有的模型都要具有很好的解释性，但如果能把模型很好地解释给同事或者客户，何乐而不为呢？

5.2.4 可扩展性

制约模型可扩展性的是模型的成本，现实中会考虑以下三方面的成本。

(1) 学习时间：也就是模型的训练时间。
(2) 得分时间：也就是模型的预测时间，给定一个数据和已经训练好的模型，新用户要多久才能得到一个预测值？
(3) 模型存储：模型运行时要占用多大的内存？

“An Emprical Comparison of Supervised Learning Algorithms”（“监督学习算法的比较研究：基于实证的观点”，http://goo.gl/bLpoea）是一篇关于算法比较的很好的文章，读一读你会学到以下几点。

- 模型的复杂程度经常与它的精度成正比。简单的模型可能具有更好的可解释性，但可能不会有令人满意的预测精度。
- 没有万能算法，每个问题都要根据其自身的特点选择最适合的模型。
- 诸多条件会限制着你对可用算法的选择，包括数据量大小，项目成本和时间成本等。

5.3 逻辑回归：一个来自M6D的真实案例研究

在 M6D，Brian 和他的团队亟须解决以下三个核心问题。

(1) 特征工程：找出最有用的特征变量并且知道如何正确处理和使用它们。
(2) 交互预测：模型需要具备迅速的用户响应能力，用户鼠标按下的一瞬间，模型要即时地预测和输出。
(3) 精准定价：若给定广告能够向目标用户展示，可以产生多大价值？

5.3.1 点击模型

对于 M6D 公司来说，工作的核心就是根据客户的要求找到广告的最佳受众。也就是说，他们需要计算在电脑前点击鼠标的你，有多大的可能点击他们所要展示的广告。那么对于这样一种商业模式，需要分析什么样的数据呢？他们应该如何使用模型来提高广告的点击度呢？

M6D 会追踪客户访问的每一个网页，但是数据科学家并不会挨个分析网页的具体内容，而只需要知道网页对应的 URL 地址。他们会将这些地址表示为一些随机的字符串，随着用户的网页访问历史不断积累，这些字符串会累积成一个字符串向量，代表用户访问的完整历史。譬如说用户 u 在一段时间内访问了四个网页，那么可以用下面的字符串表示他的 URL 访问历史：

```
u = <&ltfxyz, 123, sdqwe, 13ms&gtg>
```

其中每个子字符串都代表其访问过的某个网页的 URL。如果每个用户的访问历史都用这样的字符串向量表示，那么全部用户的访问历史可以合并成一个巨大的矩阵：矩阵的行代表用户，列代表其访问过的网站。如果矩阵的某个元素值为 1，则代表相应的用户访问过相应的网页。可以想象，这样的一个矩阵将会十分稀疏，也就是说，会有大量的 0 值存在。这是因为网站访问是一个非常个性化的行为，每个人访问过的网页都不尽相同。

如果这个数据是被用作一个分类的问题，我们至少需要一个分类变量作为被解释变量。对于点击模型来说，可以假设有一条卖鞋的广告，而我们想要分析人们是否点击了此条广告。此处的分类变量就是该条广告是否被用户点击过，它是一个二元变量值：1 代表被某用户点击过，0 则表示没有被该用户点击。当我们设定好一个被解释变量之后，加上之前已经处理好的稀疏特征矩阵，数据的准备工作就完成了。换句话说，一个**训练数据集**已经准备完毕。

接下来的任务就要交给科学家了：搭建模型、在数据集上训练模型等。第 3 章的垃圾邮件过滤例子中，训练数据集中的特征变量是单词。具体说来，那里是代表单词是否出现在某封邮件中的一个二元变量，因此当时的模型并不关心单词的真实含义。现在，你可以依赖已为垃圾邮件检测构建好的算法。

此处的情形其实十分相似，我们并不关心某个网页的真实内容，而只关心用户是否访问了该网页。因此朴素贝叶斯模型也同样适用。然而，还有另外一个模型也同样适用于该情形，那就是逻辑回归模型，它是本章的绝对主角！

一言以蔽之，逻辑回归的输出值是用户点击某广告的概率值。这和朴素贝叶斯模型的输出值如出一辙。因此，在得到模型的概率输出值之后，由你决定输出的值到底应该是 1 还是 0。此处的惯用手法是设定一个阈值（比如说 0.75），只要输出的概率值大于此阈值则输出 1，否则输出 0。这种输出方法与传统的线性回归模型有着本质的不同：传统的线性回归模型会输出任何一个介于负无穷和正无穷之间的实数值，而逻辑回归的输出值是实实在在的概率值，它位于 0 和 1 之间。因此可以说逻辑回归是为了分类模型而生的模型。

前一章已经提醒过大家，如果你生搬硬套地使用传统线性回归模型，比如说你把数据交给 R，R 会义无反顾地输出模型的结果而不管你使用的模型是否合适。线性回归模型的输出值并不是一个概率值，你可能会得到一个小于 0 的值，或者是一个大于 1 的值。

5.3.2 模型背后

逻辑回归的微妙之处就在于输出值是介于 0 和 1 之间的概率值，那么到底为什么这样呢？在这里，我们把模型背后的数学稍作呈现，希望可以加深读者对逻辑回归的认识。唯一的奥妙就在于，数据的特征矩阵被某一个神奇的函数巧妙地转换成了一个严格位于 [0, 1] 之间的数值，那么这到底是一个什么样的函数呢？从微积分的角度来看，我们需要找一个函

数，其定义域为全体实数，而值域为 [0, 1]。反逻辑函数（inverse-logit function）就是这样一个函数。下式是该函数的表达形式，图 5-2 是该函数的图像：

$$p(t) = \text{logit}^{-1}(t) = \frac{1}{1+e^{-t}} = \frac{e^t}{1+e^t}$$

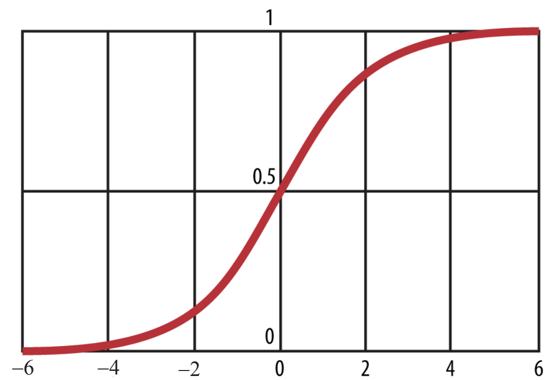


图 5-2：反逻辑函数图

逻辑函数与反逻辑函数

逻辑函数的定义域是 [0, 1]，而其值域为整个实数集。用 p 表示函数的变量，则逻辑函数可以表示为：

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \log(p) - \log(1-p)$$

顾名思义，反逻辑函数就是逻辑函数的反函数，因此其定义域为整个实数集合，而值域为 [0, 1]。

从函数的表达式来看，t 值越大 e^{-t} 值越小，因此分母越接近于 1，$p(t)$ 也越接近于 1。同样，t 值越小，则 e^{-t} 越大，因此分母的值也越大，$p(t)$ 越接近于 0。这样的一个反逻辑函数是逻辑回归的灵魂。当然，真正的逻辑回归表达式也比函数本身要复杂一点：

$$P(c_i \mid x_i) = \left[\text{logit}^{-1}(\alpha + \beta^\tau x_i)\right]^{c_i} \cdot \left[1 - \text{logit}^{-1}(\alpha + \beta^\tau x_i)\right]^{1-c_i}$$

式中，c_i 代表分类变量（被解释变量）的标签值（点击为 1，没有点击则为 0），x_i 是用户 i 的特征变量。注意，c_i 是一个二元变量值，当其取 1 时，上式可简化为：

$$P(c_i = 1 \mid x_i) = \frac{1}{1+e^{-(\alpha+\beta^\tau x_i)}} = \text{logit}^{-1}(\alpha + \beta^\tau x_i)$$

同理，当其取值为 0 时：

$$P(c_i = 0 \mid x_i) = 1 - \text{logit}^{-1}(\alpha + \beta^{\tau} x_i)$$

对上面两种简式作商，可以得到类似线性模型的表达式：

$$\log \frac{P(c_i = 1 \mid x_i)}{1 - P(c_i = 1 \mid x_i)} = \alpha + \beta^{\tau} x_i$$

也可以用刚才提到的逻辑函数来表示，那么模型的最终形式为：

$$\text{logit}(P(c_i = 1 \mid x_i)) = \alpha + \beta^{\tau} x_i$$

如果你觉得我们是在绕圈圈，那么你的感觉是对的！绕圈圈的目的是告诉你，逻辑回归的最终形式可以巧妙地转化成线性模型的形式，而模型的输出结果却被限制在了 0 和 1 之间。

有了上面的最终表达形式，我们便可以给逻辑回归一个明确的定义了：对于点击模型来说，用户 i 点击卖鞋广告的逻辑概率可以用该用户网页访问历史的特征变量（也就是之前对 URL 预处理得到的稀疏矩阵）的线性组合来表示。

模型中的 α 称作*基准值*，也就是在对用户的访问行为一无所知的情况下对用户点击广告概率的无条件猜测。从均值的含义来理解，它代表了用户群体点击广告的平均概率值。因为我们其实对此知之甚少，所以它的值一般都很小，通常近似 1%。从概率论的角度来看，它代表无条件概率值。

如果你不了解自己的特定情况，只知道基准值，那么预测值的均值只与 α 相关：

$$P(c_i = 1) = \frac{1}{1 + e^{-\alpha}}$$

而 β 是模型的斜率值，它一般是一个向量，其长度和模型的特征变量个数相同。β 向量中的每一个值都代表了相应特征变量对模型输出概率值的相对贡献程度。

5.3.3　α 和 β 的参数估计

在应用模型预测之前，首先要估计模型中的参数 α 和 β。传统线性回归中的参数估计方式并不适用于逻辑回归，因为逻辑回归的似然函数的最大化问题没有解析解，不能通过传统的求导方式解决。但是，由于其似然函数是一个典型的凸函数，因此可以用凸优化的方法找到最优解。

用 Θ 表示 α 和 β 的参数组合，L 代表模型的似然函数，则有：

$$L(\Theta \mid X_1, X_2, \cdots, X_n) = P(X \mid \Theta) = P(X_1 \mid \Theta) \cdot \cdots \cdot P(X_n \mid \Theta)$$

此处我们假设数据点 X_i 是相互独立的，$i = 1, \cdots, n$ 代表每一位用户。这里的独立性假设表明一位用户的点击行为与另外一位用户毫不相干——在这里的例子中，用户的“点击行

为”指的是用户的“点击概率”。该假设虽然并不代表真实情形，但通常来说是可以被接受的。（在上式中，我们之所以将似然函数写成概率密度函数连乘积的形式也是因为独立性假设。）

给定了数据之后，我们需要找到一组 α 和 β 最大化上面的似然函数。观察数据可知：

$$\Theta_{MLE} = argmax_{\Theta} \prod_{1}^{n} P(X_i \mid \Theta)$$

令 $P_i = 1/\left(1 + e^{-(\alpha + \beta^t x_i)}\right)$，其代表某一个观测值的概率值，那么一个数据点的概率密度 $P(X_i \mid \Theta)$ 为：

$$P_i^{c_i} \cdot (1 - p_i)^{1 - c_i}$$

假设独立性假设成立，那么参数的最大似然估计可以写作：

$$\Theta_{MLE} = argmax_{\Theta} \prod_{1}^{n} p_i^{c_i} \cdot (1 - p_i)^{1 - c_i}$$

那么现在的问题是，如何最大化似然函数呢？

对于线性回归来说，可以应用微积分中的极值定理找到能够最大化似然函数的参数组合：分别对 α 和 β 取一阶导数并令其为 0，然后检查相应的二阶偏导数是否小于 0。然而，对于逻辑回归来说这招却是行不通的。这下该如何是好呢？解决办法是有的：先把似然函数写成对数似然函数，因为对数函数是一个单调递增函数，因此最大化似然函数等同于最大化对数似然函数。然后在对数似然函数前取负号，得到负对数似然函数。这样我们就把一个最大化问题变成了一个最小化问题。取负号是因为负对数似然函数是一个凸函数，因此可以用凸优化的方法找到最小化负对数似然函数的最优解，也就是我们想要的参数组合 α 和 β。

凸优化是一个较为成熟的研究领域，有很多现成的算法可以使用，我们需要决定到底使用哪个算法。接下来我们将介绍其中两种较为常见的。通常情况下，这两种方法都可以收敛到*全局最优解*。这里的“通常情况”指的是数据的变量之间相互独立，从而保证优化过程中的海塞矩阵（Hessian matrix）满足其正定性条件。

关于最大似然估计

刚才的似然函数部分我们讲得可能太快了。如果你对最大似然估计很感兴趣，我们建议你翻一翻 Casella 和 Berge 合著的 *Statistical Inference*（《统计推断》）一书。如果你对其中涉及线性代数的部分感到吃力，我们建议你把 Gilbert Strange 教授的著作 *Linear Algebra and Its Applications*（《线性代数及其应用》）仔细研读一番。

5.3.4 牛顿法

微积分中的牛顿法是函数优化的经典方法，我们可以尝试用它找到对数似然函数的全局最优解。在微积分中，我们知道一个函数的泰勒展开式的前几项可以很好地近似该函数本身，牛顿法就是基于此原理。

具体来说，我们定义 $\nabla\Theta$ 为函数的局部梯度值（也就是在某点上的一阶导数），$\boldsymbol{H}$ 表示海塞矩阵（也就是函数的二阶导数矩阵），那么牛顿法会一步一步（迭代）地找到最优解，其迭代的步幅用 γ 表示，迭代公式为：

$$\Theta_{n+1} = \Theta_n - \gamma \boldsymbol{H}^{-1} \cdot \nabla\Theta$$

可以看出，牛顿法参数最优化的过程中，其每一步迭代的方向都与对数似然函数的曲度保持一致，也就是说每一步迭代的目的都是向极值点不断地靠近。由于式中涉及海塞矩阵（$k * k$）的求逆操作，因此牛顿法在数据量较大、变量较多的情况下（如 10 000）会遇到计算瓶颈。但是大多数情况下也不会出现变量过多的情况。

其实在实际计算中，我们很少直接对海塞矩阵求逆，而是通过解一个类似 $\boldsymbol{A}x = y$ 的线性方程组间接地找到 $\boldsymbol{A}$ 的逆（$\boldsymbol{A}^{-1}$），这通常比直接求逆矩阵要容易得多。

5.3.5 随机梯度下降法

随机梯度下降法（http://en.wikipedia.org/wiki/Stochastic_gradient_descent）是求似然函数极值的另一个算法。前面已经提过梯度，指的是函数在某一点处的一阶导数的值，也就是函数的变化率。随机梯度下降是一个顺序算法，每次只关注一个数据点，每迭代至下一个数据点都会根据所得到的信息最优地更新参数的估计值，以此类推直到穷尽所有的数据值。这相比于之前的牛顿法，优点在于它不需要对矩阵求逆。通常就这一点就可以完爆其他的算法，因为它可以有效地适用于大数据和稀疏特征矩阵的情形。因此也被类似于 Mahout（http://mahout.apache.org/）和 Vowpal Wabbit（http://hunch.net/~vw/）这样的开源机器学习软件包所重点收纳。当然，它也有缺点：它的优化效果有时候不会太好，并且十分依赖于步幅的设定。

5.3.6 操练

逻辑回归的参数估计比传统回归更适用于分类模型，但是其参数的估计方法，包括迭代加权最小二乘法和随机梯度下降等方法，都要比普通最小二乘方法复杂得多。然而，你并不需要自己编程实现这些参数的具体估计过程，类似 R 这样的统计软件已经把这些方法都收入麾下了，你所要做的只是找到适合的函数并应用到手头的数据上即可。比如说，我们手上有一个小数据集，只有 5 行观测值和 5 个 URL 标签。

```
click  url_1  url_2  url_3  url_4  url_5
1       0       0     0      1      0
1       0       1     1      0      1
0       1       0     0      1      0
1       0       0     0      0      0
1       1       0     1      0      1
```

我们把这个数据矩阵用 train 命名，那么在 R 中建立一个逻辑回归模型基本上可以用下面的一行代码解决：

```
fit <- glm(click ~ url_1 + url_2 + url_3 + url_4 + url_5,
           data = train, family = binomial(logit))
```

5.3.7　模型评价

本章一开始曾经指出，要得到一个效果良好的分类模型，数据科学家需要考虑诸多因素。其中，选择模型评价标准就是一个十分重要的因素。第 3 章和第 4 章讨论过了如何评价线性回归模型、*k* 近邻以及朴素贝叶斯模型。但是总体来说，模型的评价没有一个统一标准，它既取决于分析的数据，也和模型本身有关。因此在选择的时候需要十分谨慎。就拿逻辑回归来说，它既可以用于对二元变量的建模，也同样适用于多标签变量，因此在选择模型评价方法时，需根据不同问题采取不同方法。

首先，若将逻辑回归用在排序模型中，比如说你想将广告或商品被用户点击的概率进行排序。你可以先利用逻辑回归估算各个概率，把模型的输出结果从大到小排序。如果建模者对输出类别的相对排序感兴趣（而不是个别类的绝对概率值），那么适用的模型评价方法包括下面两个。

- 操作者特征曲线面积（ROC面积）

  前面我们提到，逻辑回归的输出值是一个概率值。如果我们想得到一个二元输出值，那么可以定义一个阈值。对于一个分类模型的分类效果，通常我们会比较关注两个指标：真阳性值（实际类别为 1，预测类别也是 1）和伪阳性值（预测类别是 1，而实际类别为 0）。操作者特征曲线（也就是 ROC 曲线）结合了阈值和两个阳性值指标，最早是被信号检测学家用来选择最佳信号的检测模型。该曲线图的纵轴代表模型的真阳性值，横轴代表模型的伪阳性值。给定一个逻辑回归模型和一个阈值，模型分类效果的真阳性值和伪阳性值都可以方便地计算出来，并可以绘制成一个二元平面上的点。阈值可以在 $-\infty$和 ∞ 之间连续变动，因此代表（真阳性值，伪阳性值）的点也可以连续变动，既而形成一条曲线。这条曲线就称作操作者特征曲线，它描绘了模型的分类效果与阈值的变动关系。该曲线到横轴之间的面积通常称作操作者特征曲线面积（AUC），它是评价一个分类模型效果的经典指标，因此也可以用来比较两个分类模型孰优孰劣。想知道更多关于 ROC 和 AUC 的细节，你可以参考 Tom Fawcett 的文章“Introduction to ROC Analysis”（“ROC 分析导论”，参见 http://goo.gl/fE8i7s）。

- 累积提升图

  在直销行业，建模者较多地使用累积提升图判断一个模型到底有没有用。模型有用的最低标准是，它的效果至少应该好于随机猜测。

在第 13 章中，我们还将用到这两种图形评价方法。

> **模型产品化的问题**
>
> 如果你想把逻辑回归产品化并应用于广告排序的实际问题上，那你应该仔细审视所用数据的生成过程。这可能说得有点抽象，让我们用一个实例来说明。譬如说，在给用户展示广告时，你将某发胶的广告放在了香体露广告的上方，然后发现发胶广告获得了更多的用户点击率，那么这究竟意味着发胶广告更受用户欢迎呢，还是因为你把它放在了更加显眼的位置呢？也就是说，广告的排列位置可能会潜在地影响广告本身的点击率，而广告本身的质量就变得难以考证了。[1]解决“干扰因子”影响的办法就是把该因子本身作为预测变量加入到模型当中。在建模过程中，干扰因子的影响十分复杂却又至关重要，我们也很难三言两语就说明白。可以想象，在谷歌甚至存在一“广告质量控制部”，专门负责研究和解决类似的问题。

其次，假设要将逻辑回归模型用于分类问题，那么我们已经知道数据的实际输出是一个二元值（0, 1），而逻辑回归的模型输出是一个介于 0 和 1 之间的概率值。为了将逻辑回归模型应用到没有标签值的样本，我们需要将逻辑回归的实际输出（一系列概率值）转换成二元值（0 和 1）。为了最小化模型的误分率，我们需要在这个转换过程中选择一个合适的阈值。比如说阈值为 0.5，那么只要预测概率值大于 0.5，预测标签值就为 1（代表被点击）；反之则为 0。模型的评价有很多指标，我们在第 3 章和第 4 章也讨论了一些。我们把这些已经讨论过的模型评价标准拿出来，再结合一些没有讨论过的标准，在这里一起呈现给大家。你会想，如此多的标准，在实际应用中到底应该选择哪一个呢？其实，每一个标准都各有侧重，对它们的选择也要因地制宜。

- 提升度

  提升度指的是分类模型的预测精度相比随机猜测模型的提升幅度。

- 准确度

  模型的准备度指的是所有被正确预测的类别（包括阳性类和阴性类）。

- 精确度

  模型的精度指的是模型的真阳性值 / 所有的阳性值。也就是说，在所有被预测为阳性的类中，其真实为阳性的比例。

注 1：在统计学中，这通常称作“干扰因子”。

- *召回率*

  召回率指的是模型的真阳性值 /（模型的真阳性值 + 模型的假阴性值）。也就是，在所有应该被预测为阳性的类中，其真实被预测为阳性的比例。

- *F得分*

  之前我们没有提到任何关于 F 得分的内容，它是精度和召回度的调和平均值，其形式为：(2× 精确度 × 召回度) / (精确度 + 召回度)。可以看出，它综合考虑到了精确度和召回度的不同特性。F 得分还有很多变种，其区别主要在于精确度和召回度的权重不同。

最后，如果我们想评价或比较模型输出的实际概率值的精确程度，可以用以下三种评价标准。

- *均方误差*

  之前关于线性回归的章节中提到过均方误差的概念，它指的是实际值和预测值之间的平均平方距离。

- *根均方误差*

  顾名思义，根均方误差就是均方误差的平方根。

- *平均绝对离差*

  与均方误差不同的是，平均绝对离差计算的是预测值和实际值之间的绝对值距离，而不是平方距离。

综合来看，AUC（接受者操作特征曲线下面积）要比提升曲线更适合用作模型比较。提升曲线的一大特征就是："基率不变性。"比如说，如果模型将用户点击率从 1% 提升到 2%，其提升率为 100%；而如果从 4% 提升到 7%，其提升率反而小于 100%。然而，很明显后者的提升效率更高。从这一点来看，AUC 要更加适用于模型的比较。

我们有必要再讨论一下输出概率模型与输出排序模型之间的区别。假设在广告点击模型中，每条广告的成本是 c 元，而如果该广告被用户点击（用 conversion 表示），可得收益为 q 元。若想获取利润，那么相应的收益必须大于成本，也就是说：

$$P(Conversion \mid X) \cdot q > c$$

可以看到式子的最左边代表用户点击某广告的概率值大小。因此，在利润要求下，能够准确估算出广告被用户点击的概率值对于保证利润有着举足轻重的作用。在这里，广告排序（谁更有可能被用户点击）的意义就相形见绌了。比如说，某三条广告的排序为 1、2、3，代表第一条广告最有可能被用户点击，其对应的概率值可能为 0.7、0.5 和 0.3。然而，即使对应的概率值只有 0.03、0.02 和 0.01，它们之间的排序关系也不会有任何改变。但是这么小的概率值对于利润的影响却是巨大的。因此，类似*均方误差*的评价标准更适合用在此处。

在模型评价中应用 A/B 测试优化法

当我们选择某个模型评价标准（比如前面提及的准确度、均方误差等），并据此建立和优化模型的时候，从根本上来说，我们是在找模型中相关参数的最优解。有时候对模型的评价适宜使用复合的标准，例如前面谈到的利润标准，而利润标准的本质其实还是模型的概率输出的准确度。因此模型本身很难直接反映我们想要的最优结果：最大化的利润。为了克服这个缺点，通常的做法是应用 A/B 测试优化法。举例来说，我们想比较两个模型哪个更优，那么我们可以随机地给两个模型组分配两组用户，并分别计算两组用户产生的利润值。这里需要注意的是，我们直接使用了我们最为关心的"利润值"指标，而并非模型准确度、均方误差这样的模型类指标，因此我们得以直接考察两组模型在创造利润上的差距，从而决定使用哪个模型。这就是统计实验设计学中的 A/B 测试优化法，我们将在第 11 章做深入探讨。

5.4 练习题

M6D 慷慨地提供了一个数据集供我们练习逻辑回归的建模、分析和评价。数据可以从 https://github.com/oreillymedia/doing_data_science 直接下载。下面的代码可作参考。

示例R代码

```
# 作者：Brian Dalessandro
# 读取数据，检查变量，创建训练和测试数据集
file <- "binary_class_dataset.txt"
set <- read.table(file, header = TRUE, sep = "\t",
                  row.names = "client_id")
names(set)

split <- .65
set["rand"] <- runif(nrow(set))
train <- set[(set$rand <= split), ]
test <- set[(set$rand > split), ]
set$Y <- set$Y_BUY

##########################################################
###########               R 函数              ##########
##########################################################

library(mgcv)

# GAM 平滑图函数
plotrel <- function(x, y, b, title) {
    # 对数据生成一个 GAM 平滑表示
    g <- gam(as.formula("y ~ x"), family = "binomial",
             data = set)
    xs <- seq(min(x), max(x), length = 200)
    p <- predict(g, newdata = data.frame(x = xs),
                 type = "response")
```

```
    # 现在得到实证分析的结果（如果结果不是离散化的，还要进行离散化操作）
    if (length(unique(x)) > b) {
        div <- floor(max(x) / b)
        x_b <- floor(x / div) * div
        c <- table(x_b, y)
    }
    else { c <- table(x, y) }
    pact <- c[ , 2]/(c[ , 1]+c[, 2])
    cnt <- c[ , 1]+c[ , 2]
    xd <- as.integer(rownames(c))
    plot(xs, p, type="l", main=title,
         ylab = "P(Conversion | Ad, X)", xlab="X")
         points(xd, pact, type="p", col="red")
    rug(x+runif(length(x)))
}


library(plyr)

# wMAE 图函数及其相关计算
getmae <- function(p, y, b, title, doplot) {
    # 将数据正则化到 [0, 1] 区间范围内
    max_p <- max(p)
    p_norm <- p / max_p
    # 细分成 b 个间隔
    bin <- max_p * floor(p_norm * b) / b
    d <- data.frame(bin, p, y)
    t <- table(bin)
    summ <- ddply(d, .(bin), summarise, mean_p = mean(p),
                  mean_y = mean(y))
    fin <- data.frame(bin = summ$bin, mean_p = summ$mean_p,
                      mean_y = summ$mean_y, t)
    # 计算 wMAE
    num = 0
    den = 0
    for (i in c(1:nrow(fin))) {
        num <- num + fin$Freq[i] * abs(fin$mean_p[i] -
                                       fin$mean_y[i])
        den <- den + fin$Freq[i]
    }
    wmae <- num / den
    if (doplot == 1) {
        plot(summ$bin, summ$mean_p, type = "p",
             main = paste(title," MAE =", wmae),
             col = "blue", ylab = "P(C | AD, X)",
             xlab = "P(C | AD, X)")
        points(summ$bin, summ$mean_y, type = "p", col = "red")
        rug(p)
    }
    return(wmae)
}


library(ROCR)
get_auc <- function(ind, y) {
```

```
    pred <- prediction(ind, y)
    perf <- performance(pred, 'auc', fpr.stop = 1)
    auc <- as.numeric(substr(slot(perf, "y.values"), 1, 8),
                      double)
    return(auc)
}


# 针对某个特征变量集合，用 X 交叉验证法评估模型表现

getxval <- function(vars, data, folds, mae_bins) {
    # 将观测值分配到相应的交叉验证分组中
    data["fold"] <- floor(runif(nrow(data)) * folds) + 1
    auc <- c()
    wmae <- c()
    fold <- c()
    # 生成一个公式（formula）对象
    f = as.formula(paste("Y", "~", paste(vars,
                   collapse = "+")))
    for (i in c(1:folds)) {
        train <- data[(data$fold != i), ]
        test <- data[(data$fold == i), ]
        mod_x <- glm(f, data=train, family = binomial(logit))
        p <- predict(mod_x, newdata = test, type = "response")
        # 计算 wMAE
        wmae <- c(wmae, getmae(p, test$Y, mae_bins,
                  "dummy", 0))
        fold <- c(fold, i)
        auc <- c(auc, get_auc(p, test$Y))
    }
    return(data.frame(fold, wmae, auc))
}

###############################################################
##########             主程序：模型与作图            ##########
###############################################################
# 现在在模型中加入所有可用变量
# 检查模型的估计参数和拟合效果
vlist      <-      c("AT_BUY_BOOLEAN",      "AT_FREQ_BUY",
"AT_FREQ_LAST24_BUY",
    "AT_FREQ_LAST24_SV", "AT_FREQ_SV", "EXPECTED_TIME_BUY",
    "EXPECTED_TIME_SV", "LAST_BUY", "LAST_SV", "num_checkins")
f = as.formula(paste("Y_BUY", "~" , paste(vlist,
               collapse = "+")))
fit <- glm(f, data = train, family = binomial(logit))
summary(fit)

# 计算每个变量对应的模型评估值

vlist      <-      c("AT_BUY_BOOLEAN",      "AT_FREQ_BUY",
"AT_FREQ_LAST24_BUY",
    "AT_FREQ_LAST24_SV", "AT_FREQ_SV", "EXPECTED_TIME_BUY",
    "EXPECTED_TIME_SV", "LAST_BUY", "LAST_SV", "num_checkins")

# 生成一个空向量用来存储模型表现评估结果
```

```
auc_mu <- c()
auc_sig <- c()
mae_mu <- c()
mae_sig <- c()

for (i in c(1:length(vlist))) {
    a <- getxval(c(vlist[i]), set, 10, 100)
    auc_mu <- c(auc_mu, mean(a$auc))
    auc_sig <- c(auc_sig, sd(a$auc))
    mae_mu <- c(mae_mu, mean(a$wmae))
    mae_sig <- c(mae_sig, sd(a$wmae))
}

univar <- data.frame(vlist, auc_mu, auc_sig, mae_mu, mae_sig)

# 画出每个变量的 MAE 图
use holdout group for evaluation
set <- read.table(file, header = TRUE, sep = "\t",
                  row.names="client_id")
names(set)

split<-.65
set["rand"] <- runif(nrow(set))
train <- set[(set$rand <= split), ]
test <- set[(set$rand > split), ]
set$Y <- set$Y_BUY

fit <- glm(Y_BUY ~ num_checkins, data = train,
           family = binomial(logit))
y <- test$Y_BUY
p <- predict(fit, newdata = test, type = "response")

getmae(p,y,50,"num_checkins",1)

# 向前贪婪选择法
rvars <- c("LAST_SV", "AT_FREQ_SV", "AT_FREQ_BUY",
    "AT_BUY_BOOLEAN", "LAST_BUY", "AT_FREQ_LAST24_SV",
    "EXPECTED_TIME_SV", "num_checkins",
    "EXPECTED_TIME_BUY", "AT_FREQ_LAST24_BUY")
# 生成一些空向量
auc_mu <- c()
auc_sig <- c()
mae_mu <- c()
mae_sig <- c()

for (i in c(1:length(rvars))) {
    vars <- rvars[1:i]
    vars
    a <- getxval(vars, set, 10, 100)
    auc_mu <- c(auc_mu, mean(a$auc))
    auc_sig <- c(auc_sig, sd(a$auc))
    mae_mu <- c(mae_mu, mean(a$wmae))
    mae_sig <- c(mae_sig, sd(a$wmae))
}
kvar<-data.frame(auc_mu, auc_sig, mae_mu, mae_sig)
```

```
# 画出三个 AUC 曲线
y <- test$Y_BUY

fit <- glm(Y_BUY~LAST_SV, data=train,
           family = binomial(logit))
p1 <- predict(fit, newdata=test, type="response")
fit <- glm(Y_BUY~LAST_BUY, data=train,
           family = binomial(logit))
p2 <- predict(fit, newdata=test, type="response")
fit <- glm(Y_BUY~num_checkins, data=train,
           family = binomial(logit))
p3 <- predict(fit, newdata=test,type="response")

pred <- prediction(p1,y)
perf1 <- performance(pred,'tpr','fpr')
pred <- prediction(p2,y)
perf2 <- performance(pred,'tpr','fpr')
pred <- prediction(p3,y)
perf3 <- performance(pred,'tpr','fpr')


plot(perf1, color="blue", main="LAST_SV (blue),
     LAST_BUY (red), num_checkins (green)")
plot(perf2, col="red", add=TRUE)
plot(perf3, col="green", add=TRUE)
```

第 6 章

时间戳数据与金融建模

本章的贡献者为来自 GetGlue 公司的 Kyle Teague 以及我们的老熟人 Cathy O'Neill。Cathy 所讲的内容涉及时间序列分析、金融建模以及 fancypants regression。但在学习这此主题之前，Kyle 将与我们一起分享他关于推荐系统方面的经验。（关于这一主题，另见第 7 章。）最后，我们会了解一下关于时间戳数据的基本知识，这与 Cathy 所要讲的内容前后呼应。

6.1　Kyle Teague与GetGlue公司

Kyle Teague 是 GetGlue 公司分管数据科学与工程的副总裁。Kyle 早先从事于电子工程领域。他认为自己之所以成为一个数据科学家，与他当年在研究室和实验室里所做的大量关于信号处理方面的工作不无关系。他从很小的时候就开始接触编程，现在他主要使用 Python 语言。

GetGlue 同样是一家总部位于纽约的初创公司，它们的主营业务是关于电影电视的内容开发。传统上，我们搜寻自己感兴趣节目的方法是参考报纸 / 电视台手册上的频道列表。这样的传统最早可以追溯到 19 世纪 50 年代。而现在，电视上有成千上万的频道以及难以计数的电视和电影节目，用传统的方法找起来似乎有点不太现实了。这就是电影电视内容开发 / 推荐所要解决的问题。

GetGlue 想要改变传统搜寻电视节目的方式，提供给人们定制化的节目推荐方案。具体来说，每当用户收看某个电视节目，他们相应地制造了一条时间戳数据，也就是说他们在某个时点进行了“收看”这一动作。当然，“收看”并不是唯一的动作，用户可以“赞”某个节目，或者对某节目加以“评论”等。

用户的这种时间戳行为会被存储在一个类似于 {用户 , 动作 , 节目} 的三元向量中。图 6-1 称作对偶图，可以用来描绘这种类型的数据。

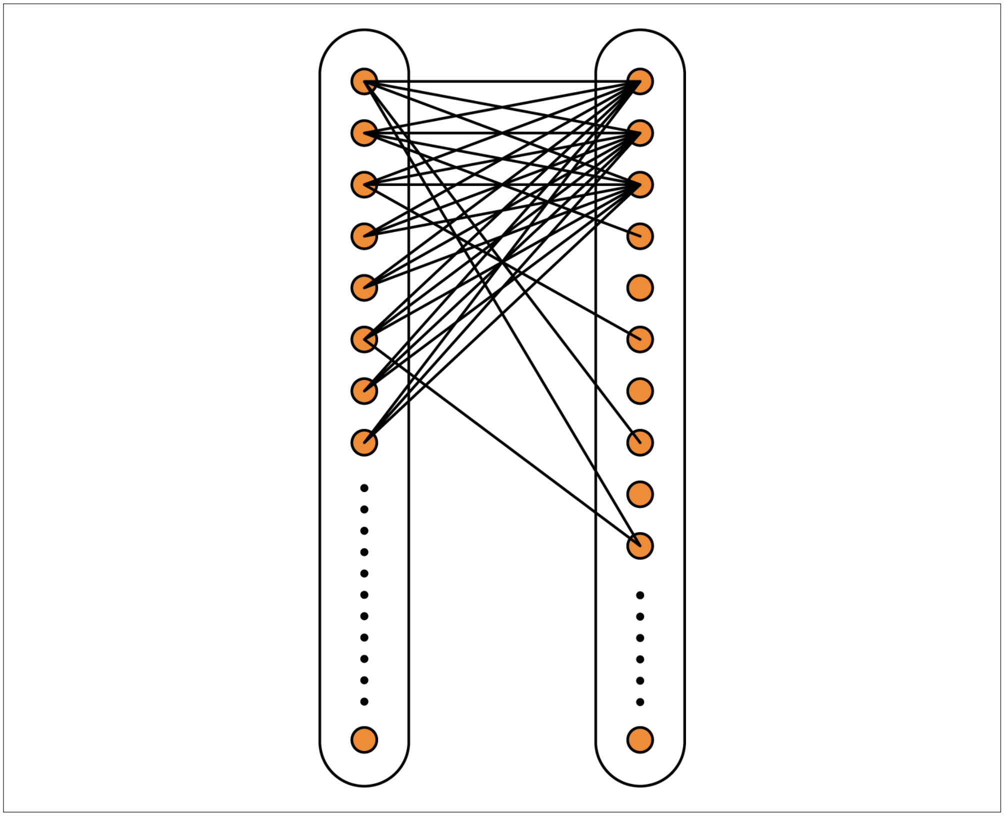


图 6-1：对偶图，左图的实心点代表用户（user），右图中的实心点代表节目（item）

图中的实心点称作“节点”，左图的节点代表用户，右图的节点代表节目；节点之间的线称作“连接线”。之所以称作“对偶图”是因为图中左右各有两个“试管”，其中的节点通过连接线相互联系。如果左图的某个用户节点与右图的某个节目节点之间有连接线，则代表该用户对该节目有过反馈。比如说，某用户点赞了某节目，则该用户和节目之间就用连接线相连。需要注意的是，“试管”内部的节点之间不存在连接线。

GetGlue 公司突破了试管内部节点直接无连接的限制，并假设内部节点也是可以相连的。比如说，用户 *A* 可以加用户 *B* 为好友，那么 *A* 节点和 *B* 节点之间就形成了一个有向的连接线，通常用一个带有箭头的直线表示。即使 *A* 与 *B* 用户直接没有互粉，GetGlue 公司也可以通过用户的资料特征猜测某两个用户可能具有类似的欣赏品味，进而把用户 *A* 推荐给用户 *B*（当然也会把用户 *B* 推荐给用户 *A*）。

刚才所讲的是用户之间的连接，而节目之间同样可以根据相似性相互连接，这项工作

通常需要人工来做，而 GetGlue 公司就雇了不少员工专门从事这项工作。比如说，电影 *TrueBlood*（《真爱如血》）和 *Buffy the Vampire Slayer*（《魔法奇兵》）从某个角度来看非常相似，那么工作人员就会在表示这两个电影的节点上画上连接线，以表示它们互相之间有相似性。但是如何确定连接线是否带有箭头，带单向箭头还是双向箭头就比较微妙了。通常的做法是按照时间先后顺序。比如说，《真爱如血》在《魔法奇兵》之前上映，那么箭头的方向就从前者指向后者。《真爱如血》当年票房一片红火，这也可能意味着《魔法奇兵》也会同样受欢迎，因为它们都是关于吸血鬼的电影。因此，这里的箭头不仅仅代表了电影内容的相似性，也可能意味着他们受欢迎程度的相似性。著名的在线电台公司 Pandora 据说也在产品中使用类似的标注法。

在这个行业，推荐的时效性是最为重要！很多的电视节目，过了这个村可能就没有这个寨了；而用户的“点赞”行为也是一闪而过，因此对数据的搜集要讲求一个字：“快”，并且每个数据点的时间戳也至关重要，这就是意味着我们要把数据的形式增加到四元：{ 用户，动作，节目，时间戳 }。记录时间戳信息可以帮助我们掌握用户喜好的传播特征，从而更好地因地、因时的为用户推荐他们最想看的电视节目。

6.2 时间戳

带有时间戳的数据是大数据时代的典型特征之一，它促进了大数据时代的到来。因为人们在使用计算过程中的每个动作其实都会被计算机精准地记录下来，尤其是这些动作发生的时点。这就意味着，即使是单个用户在一天之内也可以产生巨大的数据量。比如说，你在网站上的每次点击，或者每次使用软件，甚至是打电话，都会被计算机（或者手机 / 平板等移动设备）记录下来：包括这些动作发生的时点、持续的时间和动作的具体内容等。在软件业，新产品或者产品的新功能发布后，软件工程师一般都会在软件内部放置一段记录用户行为的代码，这段代码也是产品的一个重要组成部分，用来侦测用户对软件的使用情况以期不断改善该产品的用户体验。

比如说，在用户访问《纽约时报》的主页时，网站会记录用户被呈现了哪些文章，而用户实际点击并阅读了哪些文章等，这样跟对一个客户，会有大量的日志文件（也就是时间戳数据）产生。

下面是一段来自 GetGlue 的公司的用户原始数据：


```
{"userId": "rachelschutt", "numCheckins": "1",
"modelName": "movies", "title": "Collaborator",
"source": "http://getglue.com/stickers/tribeca_film/
collaborator_coming_soon", "numReplies": "0",
"app": "GetGlue", "lastCheckin": "true",
"timestamp": "2012-05-18T14:15:40Z",
"director": "martin donovan", "verb": "watching",
"key": "rachelschutt/2012-05-18T14:15:40Z",
```

```
"others": "97", "displayName": "Rachel Schutt",
"lastModified": "2012-05-18T14:15:43Z",
"objectKey": "movies/collaborator/martin_donovan",
"action": "watching"}
```

我们可以按照之前讨论的四元数据结构把相应的字段提取出来，{"rachelschutt", "action": "watching", "title":"Collaborator", timestamp:"2012-05-18T14:15:40Z" }就代表{用户，动作，节目，时间戳}，其中用户为 rachelschutt，动作是“观看”，节目是一个题为“合作者”（Collaborator）的文章，时间发生在 2012 年 5 月 18 日下午 2 点 15 分 40 秒。

6.2.1 探索性数据分析（EDA）

第 2 章我们已经重点提出，在建模之前最好充分地利用探索性数据分析的方法彻底了解手中的数据。这里针对用户的行为数据，我们再深入讨论一下探索性数据分析的方法。EDA 是一项富有灵活性的工作，不同的数据类型、不同的研究目的，相应都需要有不同的探索思路，因此没有统一的、纲领性的 EDA 工具，数据科学家要学会灵活应对。

对于用户行为数据分析来说，EDA 要做的第一件事就是*画出每个用户的使用行为时间序列图*。想掌握用户的整体特征，前提是做足够细致的局部分析工作。研究者要从单个用户的角度去挖掘数据所阐释的意义，从而确保所收集的数据是合情合理的，但这并不意味着你需要审查完每一个用户。

先在所有用户中取出一个样本，通常不需要太多用户，100 个即可。也许产品有上百万的用户，但为了先搞清楚状况，你得先挨个研究单一客户的使用行为。仔细审查上百万用户的数据是不可能完成的任务，同样也没有必要。通常看完 100 个用户的数据之后，你对用户整体特征应该已经了解透彻。但是，如果想更精细地研究和推断用户行为特征，100 个数据肯定不足，而且你还可能需要模型的帮助。

假设你随机选取了 100 个用户，并把他们各自用一个时间序列图表示了出来，图 6-2 展示了 4 位用户的时间序列图。

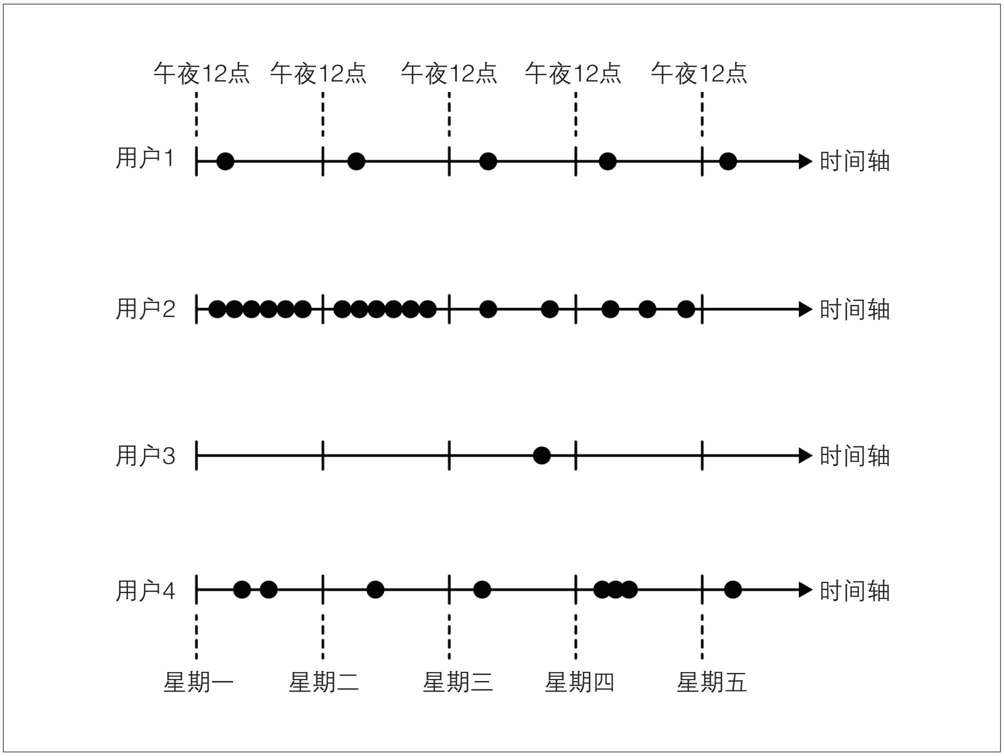


图 6-2：随机选取的四位用户的时间序列图

从这个图中，我们可以发现什么呢？很明显，用户 1 每天使用登录的时点都基本固定，而用户 2 在刚开始频繁地登录了数次，之后的登录次数迅速减少，用户 3 在整个时间段只登录了一次，因此我们可能观察更久的时间才能了解该用户的登录特征，而用户 4 每天的登录时间和次数都不太固定，这可能是一个“正常用户”，这里的“正常”与否其实相当主观，你也可以说用户 2 是正常的。

看完了 100 个用户的时间序列图之后，可以问自己下面几个问题。

- 用户的典型或者平均特征是怎样的？
- 用户之间的行为有哪些明显的差异？
- 根据用户的特征，有可能将他们分门别类吗？
- 怎样将用户之间的行为差异定量化以便后续研究？

要回答这些问题，首先我们要和原始数据打交道。原始数据与图中呈现的样子大相径庭，所以你首先要解决的问题是如何把原数据组织成有效的形态，以便可以用类似图 6-2 的时间序列图表示和研究它们。从图中可以看出，每个用户的时间戳的个数，出现的时间都是不同的，这样的数据清理起来会比较麻烦。

比如说用户的行动有 4 种可能，即“点赞”“点衰”“喜欢”以及“评论”，那么这样的用户行为怎么用类似图 6-2 的时间序列图绘制出来呢？从数据本身的角度来说，我们应该如何存储这些数据？图 6-3 提供了一种可能，相比于图 6-2，该图使用了两种颜色表示用户的“点赞”和“点衰”动作，其中红色表示前者，黑色表示后者。

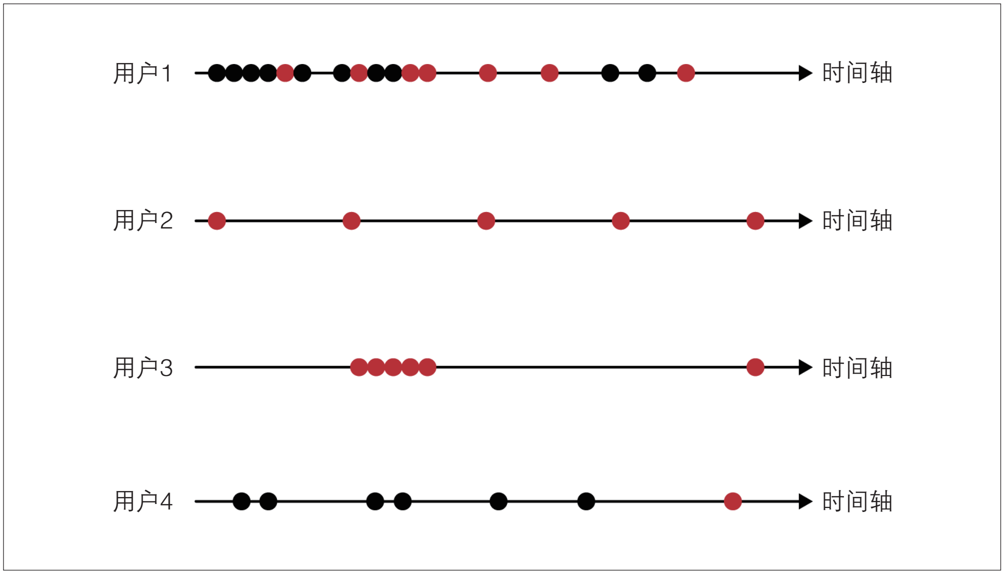


图 6-3：在用户的时间序列图中，用不用的颜色代表用户不同的动作类型。红色表示“点赞”，黑色表示“点衰”（另见彩插图 6-3）

上图中一个有趣的现象是在时间序列图的最后一个时点，所有用户都点了赞。如果发现一个类似这样的明显特征，我们最好停下来想一想到底有没有蹊跷的地方。比如说，这到底是数据本身的特征，还是系统出现了故障。如果可能是后者，我们如何确定它属于系统故障。像这种用户群体同时发生的动作在其他地方还有没有出现过？黑色的节点是否普遍要多于红色节点？用户是偏向于喜欢“点赞”还是“点衰”？是否某些用户群倾向于“点赞”而一些用户群则倾向于“点衰”，而另外一些用户群则更倾向于属于“混合型”用户？怎么定义用户的“混合型”行为？可见，能引起我们兴趣的问题很多。

完成单一用户的核查之后，下一步就要想一想怎么对用户进行汇总归类。比如说，图 6-4 就是一个汇总的时间序列图，图的最下方横轴表示时间，纵轴表示所有用户的总计数。

现在面对的是全体用户的数据，我们选择了一个汇总统计量以研究总体用户的平均行为特征。对于时间戳数据来说，即便是汇总统计量的选取也不会那么容易。因为用户可以随心所愿地登录，因此我们到底是汇总用户数还是用户的登录次数呢？又比如说，在某个特定的时间段内，是汇总用户的总行为数还是汇总进行了某项行为的总用户数？这样的抉择问题在时间戳数据处理和分析中十分常见。

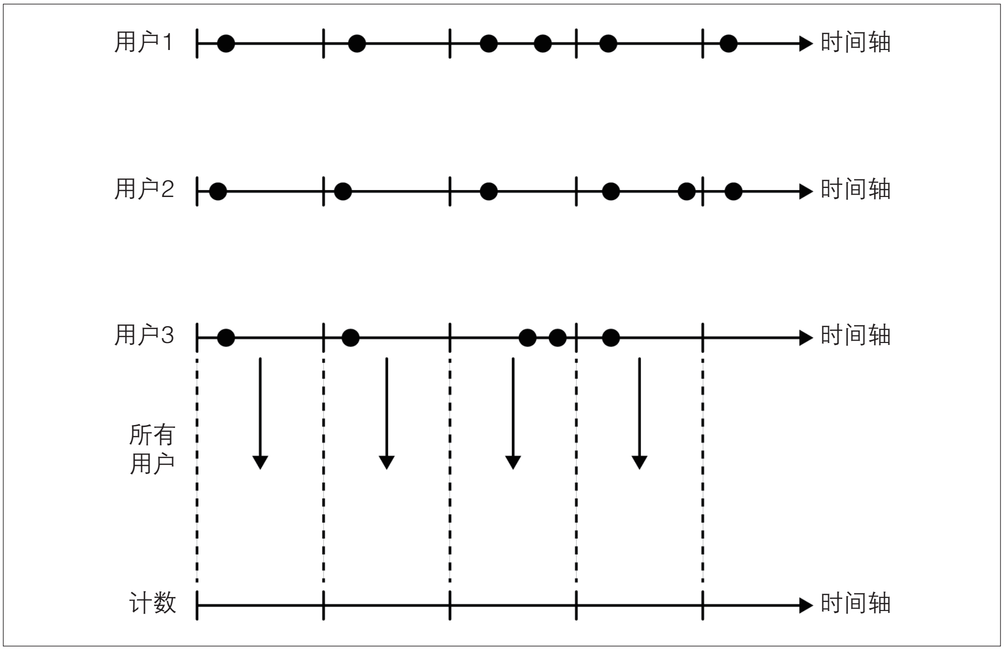


图 6-4：用户汇总时间序列图

现在让我们再回到时间序列图的本身，问自己一个更为基本的问题：这个图横轴的时间到底代表什么？是以秒计算，还是以天计算？如果选择了以天计算，那原因为何？图中是否有明显的趋势或者波动特征，波动特征是否明显掩盖了趋势特征？类似的问题还有：我们的用户是否来自同一个时区？比如说，用户 1 来自纽约而用户 2 来自伦敦，那么纽约的早晨 7 点则意味着伦敦的中饭时间。如果我们的基准时区是纽约，并总结说：从图中看来，有 30 000 个用户在早晨 7 点对某电影点了赞；如果有很多其他时区的用户，那么这样的总结就会相当不准确，甚至于歪曲了事实。对于这种情形我们又该如何处理呢？此时可以根据所有用户的时区将他们的时间戳用一个基本时区（比如说，纽约时间）来表示。类似这样的问题在数据处理中还有很多，我们需要注意分辨、分析，并根据问题的实际特征加以更正。

时间戳数据数据处理的噩梦

我们也不想骗你，时间戳数据可能是你见过的最难搞定的数据类型，时区就是一个大问题。通常的做法是把所有时间点都转换成一个基准时间（参见 http://www.epochconverter.com/）后再做分析。

我们可以做的事情还有很多，比如不同的行文类别可以用不同的图示方法表示出来，某些行为类别还可以合在一起构成新的分类。这些进一步的分析往往可以给数据科学家带来有关数据的不同方面和不同角度的有价值信息。

6.2.2 指标和新变量

探索完数据之后，我们可能已经对数据有了一定程度的认识，接下来的任务就是构建指标 / 变量了。比如说，用户点赞的次数、用户第一次点赞的时间等都可以作为模型的变量。即使是构建一些简单的二元变量也可能十分有用，比如用户“至少点赞了一次”等。构建指标和新变量的主要目的是方便我们对用户进行比较，换句话说，就是计算用户之间的相似度和相异度。在构建新指标和变量的时候，大致有两个方向可以走：一个是以用户为基准，汇总行为数；一种是以行为为基准，计算涉及的用户数。

因此，我们可以构建无数个指标或者新变量，上限就是你的想象力。但是，在构建新变量的时候，一个最基本的原则是：这些指标和变量的含义要清晰，要对理解数据有所帮助。

6.2.3 下一步怎么做

在完成探索性数据分析，并构造了一系列的新指标和变量之后，下一步当然就是建模，构思和实现算法，深入地分析数据了！但具体要怎么做还是要根据问题而定，我们这里呈现几个例子：

我们可以建立时间序列模型，并用作预测。比如说较为常见的自回归模型就是时间序列模型的一种，在下一节的金融建模中我们会详细讲解。在时间序列分析中，如果某个建模对象对时间较为敏感（比如，市场总是时涨时跌，与时间的关系密切。时间在市场分析中就扮演着极为重要的角色），这样我们就可以根据建模对象的时间特征预测它下一步会如何表现。

我们也可以做聚类分析（第 3 章已经详细讨论了聚类分析的内容）。对于用户数据的聚类分析，其关键点也同样在于如何定义用户之间的相似性。

或者我们会想要构建一个用户行为预警系统：比如说某个用户的行为特征发生了异常就自动给管理人员发送警报。当然，这里我们首先需要定义何为“异常”行为？

我们也可以做一个事件监测系统，也叫作“变点分析”。也就是说我们可以根据用户的行为特征判断在某个时点发生了“不寻常事件”。更深入的问题还包括：用户的何种行为会触发类似的事件？用户的行为与该事件的发生有无因果关系等？不可否认的是，因果关系的研究难度会比较大，它需要涉及类似上一章的 A/B 实验的内容。还有一个终极目标，我们可能会想做一个推荐系统，给用户推荐他们想要的电视节目。虽然对于我们个人来说不太现实，但这正是 GetGlue 在做的事。

> **历史背景**
>
> 时间戳数据其实并不新颖，时间序列分析本身也不是一个新兴学科。对时间序列分析感兴趣的读者可以阅读 James D. Hamilton 的 *Time Series Analysis*（《时间序列分析》）一书。在时间序列研究伊始，数据量一般都比较小，数据发生的频率也不高：通常是日度或者月度数据。比较高频的数据最早出现，是在金融学研究中的股票数据、信用卡的交易数据、电话的通讯记录数据、图书馆的图书借阅数据等。
>
> 时间戳数据的形态正在发生着日新月异的变化。这主要体现在：首先，得益于移动设备的迅猛发展，对人们日常行为的记录变得难以置信的简单和直接。其次，时间戳具有相当的精确性，因为这些数据是由设备本身记录的，而不是由人自己汇报的。众所周知，机器不会撒谎，但个人汇报的数据却并不可靠。最后，随着计算能力的迅猛发展，机器可以储存大规模的数据，处理时间也相应较快。

6.3 轮到Cathy O'Neill了

Cathy 是我们的老熟人了，她的数据科学知识构成可见图 6-5。

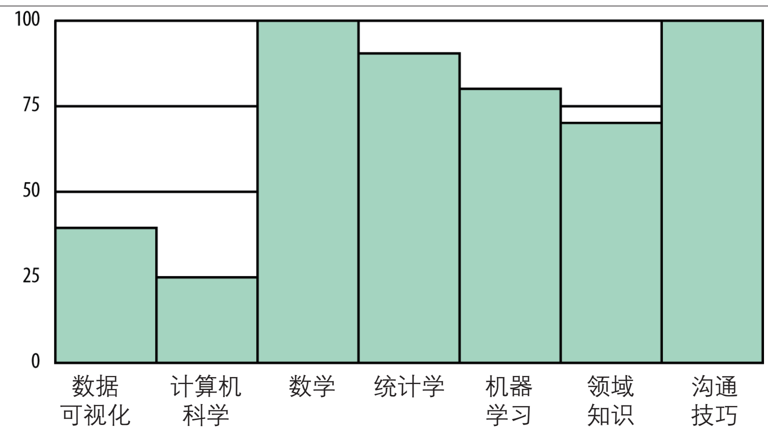


图 6-5：Cathy 的数据科学知识构成图

CS（计算机科学）是 Cathy 的明显弱项，不过她的 Python 编程技巧十分娴熟，可以轻松抓取和解析想要分析的数据，建立模型并使用 matplotlib 绘制好看的图形。然而对于使用 Java 进行 map-reducer 操作，Cathy 就不太擅长了。她也同样不太擅长数据可视化，不过她很善于演讲以及与人交流。

6.4 思维实验

如果你选择忽视数据中内含的时间戳信息，你会失去什么？

答案是，你将很难梳理出数据中内含的因果关系，因为时间往往在因果关系中扮演着至关重要的角色。

但是如果我们稍微改变一下刚才的问题，假设你并没有具体的绝对的时间戳信息，但是还是收集了一下“相对时间戳”信息，比如说“离上次用户最后登录的时间”，或者“离上次用户点击的时间”等。

答案是：即使有“相对时间戳”信息也于事无补。诸如数据中的趋势和季节性特征等都会被“相对时间”忽视。以之前的胰岛素数据为例，你可能会发现在注射胰岛素 15 分钟后，血糖会持续降低，这是“相对时间戳”会告诉你的信息。但是如果只有这个“相对时间”而没有具体的时间戳信息，你可能会忽视这样一个长期趋势：在过去的几个月中，你的血糖一直在持续走高。同样对于图 6-6，因为每个时点的绝对时间戳都被详细地记录了下来，因此可以看到有明显的季节波动趋势。如果只有“相对时间戳”数据，你很容易忽略了季节性特征。对于金融数据来说，如果想通过观测序列的走势找到合适的买入 / 卖出点，记录具体的时间戳信息更显得极为重要。

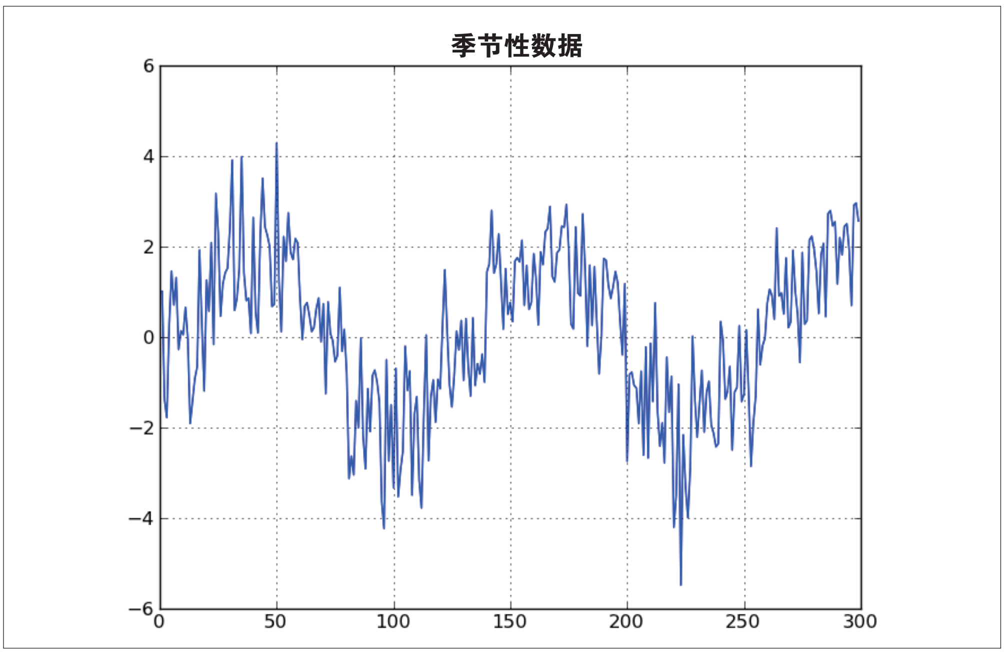


图 6-6：如果不记录具体的时间戳信息，像本图中的季节性特征是无从发现的

6.5　金融建模

数据科学家的头衔出现之前，在金融领域“金融分析师”这一职业已经存在了许久。从工作性质上来看，金融分析师与数据科学家既相似又大有不同。其中一点就是，金融分析师

十分在意时间戳数据，甚至有点迷恋，他们甚至不太关心为什么数据会是这个模样，而只关心数据发生的时间点。

金融建模是一个内涵丰盈的研究领域，我们接下来就感受一下它的魅力所在。

6.5.1 样本期内外以及因果关系

首先需要定义何为“样本期内”，何为“样本期外”，对于金融建模这是一个基本的概念区分。样本期内，顾名思义就是在观测期范围内的样本数据，而样本期外与我们前面章节所讨论过的“测试数据”有所关联，但又有本质的不用。测试数据本质上还是在观测期内，只是将观测器划分成了训练数据和测试数据两个部分，一部分用来训练模型，另一部分用来验证和测试模式。而金融建模中的样本期外的数据严格发生在建模之后，对于样本期内的数据来说，它们是未观测到的数据。其目的是用来检验金融模型在未观测数据上的真实预测表现。

此外，样本外分析的次数甚至都应该加以严格限制。因为，每一次分析，我们都将手头的样本内数据学习了一遍，如果样本外分析的次数过多，即使是研究的问题稍有不同，抑或是不同的模型，我们在潜意识里都有可能不知不觉地得到过拟合的结果。

其次，当执行*因果关系建模*的时候（这与统计学家所说的*因果关系*有所不同），我们必须多加小心。核心原则可以一语概括：*永远不要拿未来的数据去预测未来！*听起来似乎十分拗口。换句话说，意思也就是，预测样本期外的数据时只能使用样本期内的数据，因为样本期外对于样本期内来说是尚未观测到的。其实，更加严格的来说，即使是现时的数据，如果你还没有搜集到，也不能用在对未来的预测上。一语以蔽之：你只能使用样本期内已经观测到的数据！稍后我们会讲到一个涉及政府部门的时间序列数据。

同样，因为时间戳的存在，在给定一个训练数据集后我们难以确定何为模型的“最佳估计参数”。因为从数据的第一个时间戳开始估计，每增加一个时间戳，模型的估计都会发生相应的改变，直到估计到最后一个时间戳。也就是说，我们不会得到一个“最佳估计值”，而是得到一系列随着时间推进而演化的估计序列。

这类估计方法的有用之处在于，我们可以借此检验模型估计的稳定性。譬如说，如果某个参数的估计值的正负号在估计序列中前前后后变化了十几次，那么我们可以认为该参数的估计相当不稳定，而索性将其从模型中赶出去。当然，具体要不要把它赶出去还要看数据和模型的具体含义，也许它是一个天生就好动的参数，但在模型中却扮演着十分重要的角色。

图 6-7 用红色线将样本期内与样本期外分离开来，它所要表达的含义是：样本期内永远发生在样本期外之前（以避免因果关系问题）。

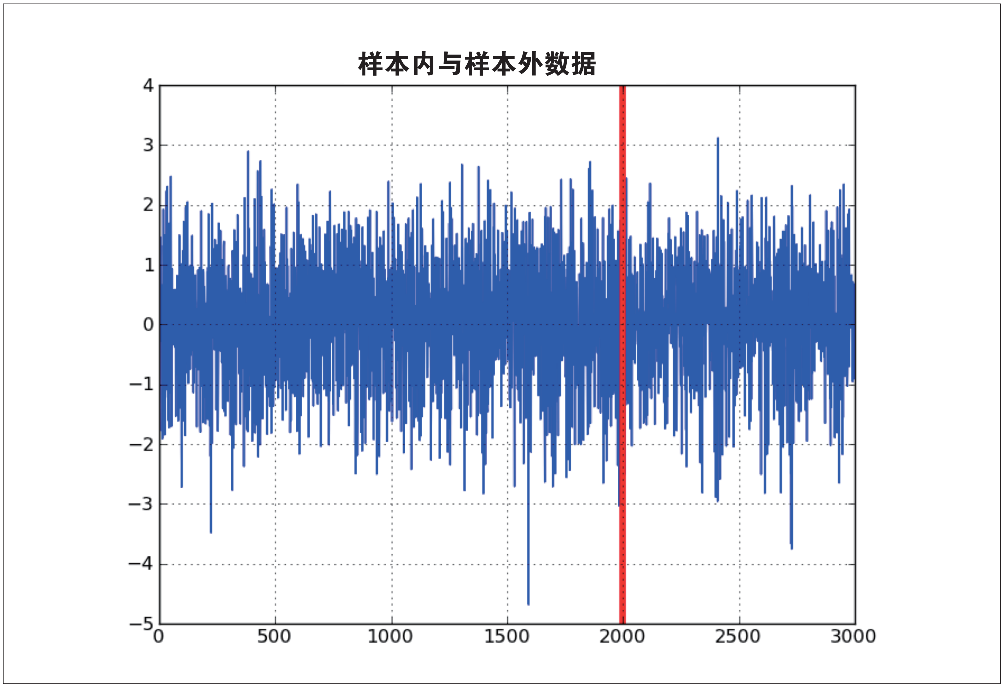


图 6-7：样本期内的数据永远发生在样本期外数据之前，红色线代表了模型建立的时点（另见彩插图 6-7）

关于因果关系建模，以及样本内与样本外的关系，在建模分析中很重要，在产品代码中也应该得到一致性的体现。我们在建模的时候——无论是在训练数据上还是在做样本外模拟——都假设模型可以用在之后的实际产品当中，并发挥相应的效用。当然，模型的训练和拟合用的是样本内的数据，因此样本内的模型表现自然要好于它之后在产品中的表现。

另一种说法是，一旦建好了模型并用在产品中之后，预测和决策只能建立在预测期之前的可得数据上，这是因果建模的核心准则。然而，一旦搜集到新数据，我们可以理所当然地把新数据加入模型，更新模型的参数估计并既而更新模型的预测值。因此，随着时间的推进，模型也理应被不断地更新、优化并变得越来越好。

6.5.2 金融数据处理

在实际建模之前，通常要对数据进行预处理和分析：比如，做对数变换，计算均值和方差等。这是很好的习惯，也称作“预建模”。

数据变换

人们对于正常数据的分布的总体印象来自于正态分布，并认为数据差不多都应该是对称的钟形分布，而许多模型恰好也假设数据服从某种形式的正态分布。然而，金融数据往往会表现“失常”，它们的分布往往既不对称，也明显不是钟形分布（比如说，分布过尖），因此需要对数据进行变换，让它们的形态回归“正常”。下面是一些常用的变换方法。

- 将数据减去其均值并除以其标准差，变换后的数值严格位于 0 到 1 之间。这称作标准化变换。
- 在标准化变换中，也可以除以数据的最大值。
- 对数变换：对变量的每个观测值取对数。
- 将数据从小到大等分成 5 个组，每组的观测值数量相同。
- 将数据变换成二元变量（0 或者 1）。最常见的变换方法我们之前有过论及，可以取一个阈值，如果变量的值大于该取值则变换为 1，否则变为 0。

一旦我们估计出了数据的均值 $\bar{y}$ 和方差 σ_y^2，新观测到的数据点也可以被标准化。如果原始变量服从一个正态分布，那么标准化后的变量严格服从标准正态分布：一个均值为 0，标准差为 1 的正态分布。标准化变换可以表示为：

$$y_1 \mapsto \frac{y - \bar{y}}{\sigma_y}$$

数据预处理的方式多种多样，具体要怎么做要取决于你的具体情况和研究目的以及之后要用的一些子模型。比如说，有时候我们可能更关心数据的相对变动，也就是说，在 t 时点，我们想知道 y_t 与上一时点的观测值 y_{t-1} 有何区别；或者，我们希望通过一个子模型（比如一个单序列的回归模型等）找到 y_{t-1} 的哪部分更适合用来预测 y_t，那么这个时候它们的差值 $y_t - y_{t-1}$ 可能就是我们更加关心的变量。[1]

数据变化的方法同样很多，到底选择何种变换也同样取决于数据本身以及研究目的。需要铭记于心的是，时间序列的观测值是有序的，时间上的顺序代表着可能的因果关系，因此当你在训练模型以及引入新数据更新模型时一定要遵循数据的先后关系。永远不要在建模中作弊，不要使用还没有发生的样本外数据。

一个典型的例子是有关在标准化操作中均值的选用：一定不要使用整个时间段的数据均值。对于时间序列数据来说，通常要计算“移动均值”，也就是你在某个时点 t 上所知道的局部均值，在标准化时也应该使用该局部均值，而非全局均值。[2]

注 1：这种差分操作在金融建模的很多场合都十分常见。

注 2：也就是说随着时间的推移，序列的均值也在相应地发生变化。这样做的原因在于时间序列数据的变动性大，不确定性高，单一的全局均值不能体现序列的变动特征。

如果不这样做，可能会导致十分危险的结果。举个例子来说，假设数据中间段发生了市场崩溃[3]，这样的极端情况虽然只是偶有发生，但仍会对整个序列的均值和方差产生巨大影响。如果在均值的估计中只采用上文所说的全局均值，模型的效果从整体上来看会显得异常得好：我们似乎可以在市场崩溃之前就预测到它的到来了。这样的伪因果估计只是从形式上提升了模型的预测能力，或者让一个原本很差的模型变得看起来还不错（甚至，让一个根本没有价值的模型变得价值连城）。[4]

6.5.3 对数收益率

金融学中，收益率观测的标准频率以日计算，市场十分关注收益率的日度变化值。每天市场都会开盘和收盘，那么计算收益率的方法可以基于开盘情况也可以基于收盘情况。以开盘为例，我们可以在周一和周二的早晨记录下两天各自的开盘价格，并用周二的价格减去周一的价格即得到周二的收益率。但是，通常的做法还是以收盘价格为基准计算收益率。

相比于百分比收益率，我们用得更多的是*对数收益率*：假设第 t 日的收盘价格为 F_t，那么该日的对数收益率可以表示为 $\log(F_t / F_{t-1})$，相应的百分比收益率定义为 $100((F_t / F_{t-1})_{-1})$，如果去掉百分比收益率定义中的 100 则得到*绝对百分比收益率*。

相比于绝对百分比收益率，对数收益率具有可加性。也就是说，5 日的对数收益率等于 5 天内每日对数收益率之和。可加性带来了巨大的计算便利性，也是通常人们更倾向于使用对数收益率而不是绝对百分比收益率的重要原因。

对数收益率另外一条有趣的性质是“对称性”。举例来说，一只股票的价格为 2，其先降低 50% 则价格变为 1（绝对百分比收益为 – 0.5），那么如果该股票的价格从 1 回到 2，其百分比收益率为 100%，绝对百分比收益为 1，而 – 0.5 和 1 是不对称的。如果使用对数收益率，则 $\log(0.5) = -0.301$ 与 $\log(2) = 0.301$ 是严格对称的。

然而如果数据的频率较高，比如说日度数据，或者更高频的分钟数据封，其百分比收益率与对数收益率之间的差距非常小。这个很容易证明：令 $x = F_t/F_{t-1}$，则百分比收益率为 $x - 1$ 而对数收益率为 $\log(x)$。对 $\log(x)$ 应用泰勒展开式，可以得到：

$$\log(x) = \sum_n \frac{(x-1)^n}{n} = (x-1) + (x-1)^2/2 + \cdots$$

因此对数收益率始终大于百分比收益率，但是只要 x 的值非常小，那么上式中右项从平方项开始会越来越小，甚至可以忽略不计。对于高频数据来说，$x - 1$ 的值通常都很小，满足忽略不计的条件。因此，对于高频数据，可以用对数收益率很好的近似百分比收益率，并且可以保留对数收益率可加性和对称性的优良特性。

注 3：金融市场的崩溃反映在时间序列上就是金融产品价格的极速下跌。

注 4：在统计学中，均值是一个典型的“不稳健”统计量，它很容易受到极端值的影响，如果一个序列中存在一两个极端值，它们可以显著地影响全局均值的大小。

图 6-8 的直线为百分比收益率，曲线为对数收益率，可以看出在 $x=1$ 附近，两条线十分接近。

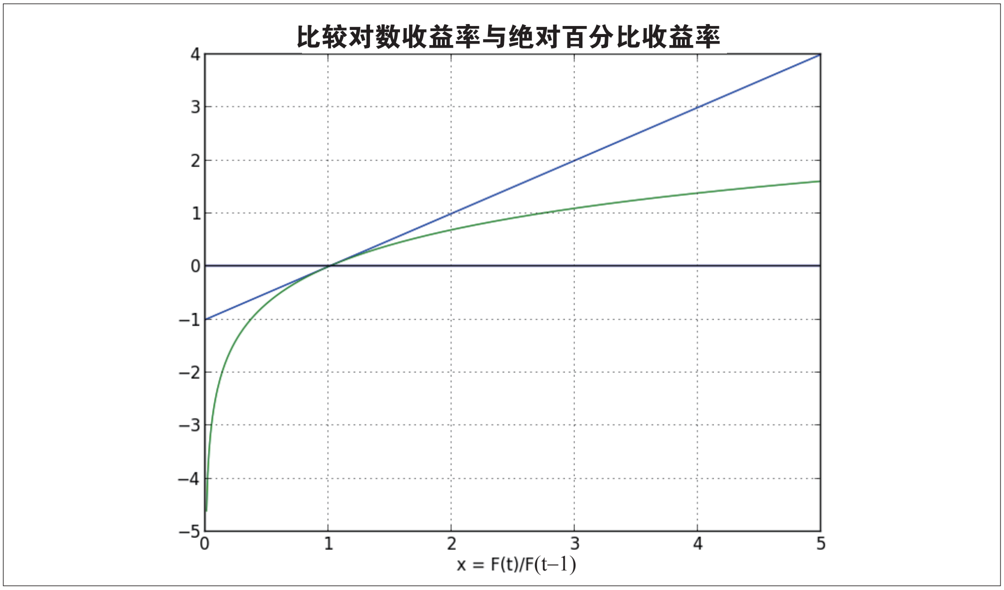


图 6-8：对数和绝对百分比收益率曲线对比图（另见彩插图 6-8）

6.5.4　实例：标准普尔指数

图 6-9 是标准普尔指数从 2001 年到 2012 年的收盘价格指数时间序列图，图 6-10 为其对应的对数收益率时间序列图。

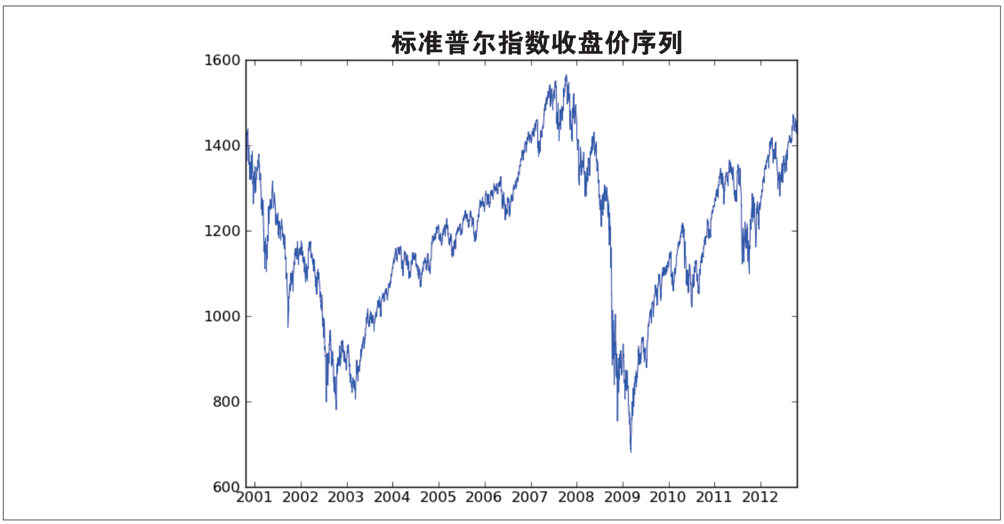


图 6-9：标准普尔指数收盘价格图

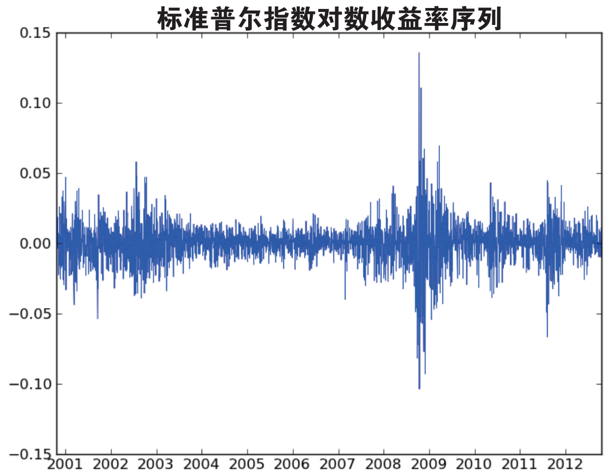


图 6-10：标准普尔指数对数收益率图

图 6-9 看起来还似乎有规律可循，图 6-10 看起来则是一团乱麻。这是金融时间序列中特有的波动率特征，2008 年底的金融危机带来的巨大波动率在图中表现得十分明显。如果应用之前提到的标准化数据变换操作，则可以有效地压缩对数收益率序列的波动率。图 6-11 就是标准化之后的对数收益率，可以看到，波动率在整个观测区间内都十分一致，没有出现图 6-10 中那样的巨大波动。

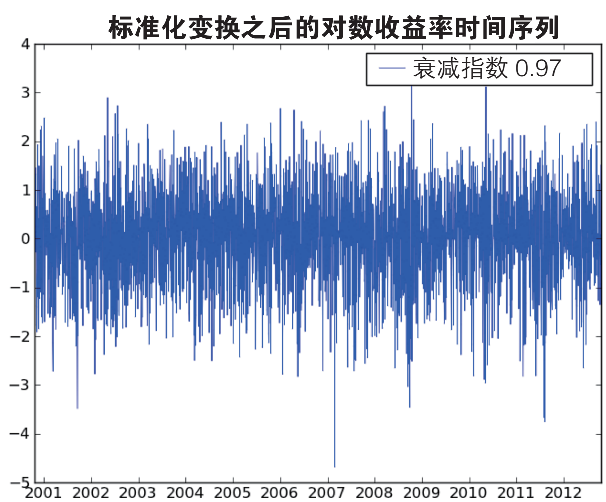


图 6-11：标准化变换之后的对数收益率时间序列图

6.5.5 如何衡量波动率

定义好收益率之后，我们可以计算收益率的变化程度，也就是收益率的波动率。通常我们用收益率样本的标准差作为*波动率估计*。

波动率计算的关键问题在于样本回溯窗口期的设定，也就是说对于某个时点 t 的波动率，适合用多少期之前的样本来估计。回溯窗口期越长，能够用到的信息越多；窗口期越短，波动率对新样本的加入则可能更加敏感。对于窗口期，试想一下市场中发生了一个大事件，比如说金融危机，那么该事件的影响需要多长时间才能消逝？这么一段时间就类似于窗口期。这个例子虽然还是十分模糊，但窗口期大概就是那么回事。对于一个大事件来说，窗口期明显要大于一周或者一个月，但超过一个季度的窗口期则比较少见。当然，这要取决于事件本身影响力的大小。

其次，在计算收益率时，对于窗口期内的数据点，我们还要决定如何确定每个数据各自的权重大小。如果是严格地滚动窗口，也就是说窗口期内的所有数据都被赋予相同的权重，窗口期之前的数据则不会被考虑进来。但这种等权重的计算方法与通常事件的演变方式有所出入。这类似于 k 近邻模型，在窗口期内离计算时点越近的点往往对其影响更大，因此在计算波动率时权重的赋值可以根据时间的差距逐渐递减。

这种逐渐递减的计算方法也称作“指数平滑法”：离得越远的数据其权重会指数式得递减。这里的指数应该是一个小于 1 大于 0 的小数。比如说如果指数值为 0.97，那么第五天前数据的权重就是 0.97^5。这里的权重是绝对权重，在实际计算时还要除以权重值的总和以计算出每个时点上数据的相对权重。如果权重严格地按照指数递减，我们可以相应地找到窗口期内的“半衰期”对应的时点：也就是在这一点上，其对计算时点事件的影响已经下降了 50%。对于 0.97 来说，其半衰期为 $-\ln(2)/\ln(0.97) = 23$。也就是说，需要经过 23 个时点，计算时点事件的影响会下降一半。

确定了窗口期和平滑指数值之后，对窗口期内的每个收益率值取平方，乘以其对应的相对权重后相加取和（平方和项），再取其平方根就得到了波动率的估计值。

注意我们刚刚给大家的公式，其计算过程包含了所有之前的收益率的值，即便权重值在指数递减，这样的计算很可能还是没有止境的，这里有一个实用的小技巧：如果窗口期很大，那么只需要计算一个波动率对应的平方和项。新时点的波动率的计算，只需要将新时点对应的收益率取平方，加入之前的平方和项即可。当然，在加入平方和项之前应该乘以其对应的权重。

我们来详细地解释一下。因为所有的权重都是指数幂的形式，那么所有窗口期内收益率权重的和也称作几何平均值，如果窗口期足够大，该值为 $1/(1-s)$，其中 s 为指数值。当前时点收益率的波动率的估计值可以表示为：

$$V_{old} = (1-s) \cdot \sum_i r_i^2 s^i$$

假设在新时点的收益率为 r_0，那么在该时点的波动率估计量为：

$$V_{new} = s \cdot V_{old} + (1-s) \cdot r_0^2$$

我们说过，波动率的估计量通常用窗口期数据的标准差来表示，而标准差的计算需要减去序列的均值。而我们刚才展示的计算过程似乎没有任何涉及均值的计算项。那是因为我们已经假定数据已经做了中心化操作，因此均值为 0；或者我们处理的是类似日度数据等高频数据，其均值通常非常接近 0。当我们计算低频数据，比如季度数据或者年度数据的波动率时，那么毫无疑问我们应该把均值的影响考虑进来。

关于指数平滑法

对于将大量数据压缩成一个估计值的问题，指数平滑法非常好用。因为，估计值的更新非常容易，我们不需要每次都重新计算一遍。

当然，用多大的指数值对于指数平滑的效果有直接影响，这里可以参考图 6-12。



图 6-12：标准普尔指数的波动率的指数平滑估计：使用了三个不同大小的指数值（另见彩插图 6-12）

其实，指数值的确定方法除了人为主观设定以外，也可以采取最小化风险函数的方法。[5]

6.5.6 指数平滑法

我们刚才用标准普尔指数的例子讨论了如何计算其收益率的波动率。这里我们更详细地介绍它的数学模型。

对于 t 时点的估计值我们假设为 E_t，设定 s 为衰减参数（其通常等于 1 – 指数，因此是一个很小的值）。指数平滑的基本思想是，离观测时点越近的值其权重越大，越远的值其权重越小，因此 E_t 的指数平滑公式为：

$$E_t = s \cdot E_{t-1} + (1 - s) \cdot e_t$$

其中 e_t 为在 t 时点的新观测值。

> **估计的可加性**
>
> 我们要求每个估计值都具备可加性（对于前面提高的滚动估计来说，方差是可加的，而标准差是非可加的）。如果我们最终想要的估计值是一个加权平均值，那么加权值的分母项与分子项都需要具备可加性。

指数平滑法固然有效，但是在实际应用时还是应该小心谨慎一点儿，尤其是在数据样本量比较小的时候（比如样本量只有几百或者更少时）。与其用一个固定的平滑参数，一种更加保险的方法是使用变动的平滑参数（其实是平滑参数的倒数，但由于数与其倒数是一一对应的，所以这里不做严格区分）。可以把平滑参数的倒数与序列的半衰期的概念联系起来，刚开始半衰期为 1，然后慢慢变长，直到逼近其真实的半衰期值 $N = 1/S$。因此，在每一个时点 t，我们可以得到一系列的 e_t 值，它们构成一个向量 v。用程序可以表示如下：

```
true_N = N
this_N_est = 1.0
this_E = 0.0
for e_t in v:
    this_E = this_E * (1-1/this_N_est) + e_t * (1/this_N_est)
    this_N_est = this_N_est*(1-1/true_N) + N * (1/true_N)
```

6.5.7 金融模型的反馈

定量分析的研究人员应该始终铭记于心的是，模型的预测效果市场会很快地回应。也就是说，就算模型能够通过分析数据在市场中赚钱，其效用迟早也会消失。这通常称作：市场的学习能力。

注 5：在工业界，通常人们都会采取后者，因为其更加客观，可以消除主观因素的影响。但是如何计算风险函数同样是个十分主观的问题。

以股票为例，如果模型告诉你市场价格将在未来的某个时点上升，那么你可能会做的就是在现在的时点买入该股票，然后等待未来的时点到来时把手里的股票卖出去以获取价格差的利润。但是试想一下，你在现时点的买入操作其实已经对市场带来影响，告诉市场参与者你预期它价格未来将会走高，这样或会改变其他投资者的决策，从而使得你所预期的信号对股票价格的影响力下降。当然，如果买入量很小，你的买入操作对市场的影响可能微乎其微。但是如果模型一直都表现良好，你的买入量会越来越大，最后你的买入操作可能会对市场产生切实的影响。因为如果你的模型一直预测良好，你的大量买入操作会给其他投资者同样的价格暗示，这会削弱你自己的盈利能力。因此在市场中摸爬滚打其实是在锻炼自己对入仓量的精准把握。

这就是市场的学习过程，每个人都在做出预判，每个人都在看别人如何预判，而市场会做出它最平衡的决定。这样的学习能力所导致的直接后果就是市场的遗忘性，即现存信号对未来市场走势的指导能力很弱。19 世纪 70 年代一次事件的影响会在很短的时间内被市场所预判、理解并消化，我们在 2014 年是不可能看到任何该事件还可能留存的影响。

确切地说，现今市场每日的记忆能力只有 3%。也就是说如果模型的预测值与真实值的相关系数大于 3%，说明你的模型很可能已经战胜了市场，你可以从中获利。3% 也同样意味着，市场中的可利用信息微乎其微，其绝大多数只能算是噪声。因此，如果模型足够出色，并且交易成本足够小，你才有可能从市场中获利。

因此，机器学习模型中的很多用来衡量模型好坏的标准，比如模型的精确度、准确度等，都不太适用于衡量一个金融模型的优劣。

抛去模型准确度这样的传统测度指标，我们更经常使用的是模型的 PnL 值，PnL 指的是模型所获取的实际收益与实际损失的差值，代表了模型的日度表现变动情况（注意，是差值而不是比例值），从计算上来看就是今天的表现值减去昨天的表现值。图 6-13 展示了两个模型的累积 PnL 值。

PnL 同样可以应用到其他的很多场合用来评价模型的好坏——基本思想就是画出去均值的预测值以及去均值的实际值的累积和。（去均值也就是将序列的每一个观测值减去其序列均值后得到的序列值。）也就是说，我们想知道所有模型是否一致性地好于那个“最愚蠢的模型”：也就是用均值预测一切的模型。[6]

如果模型的 PnL 图看起来有一直向右上方前进，则代表的模型一直表现良好。如果走势太参差不齐，你的模型可能有未知的不稳定性。

注 6：从 PnL 图中我们可以解读出当前使用的模型在样本期内是否一致地优于市场的平均表现。

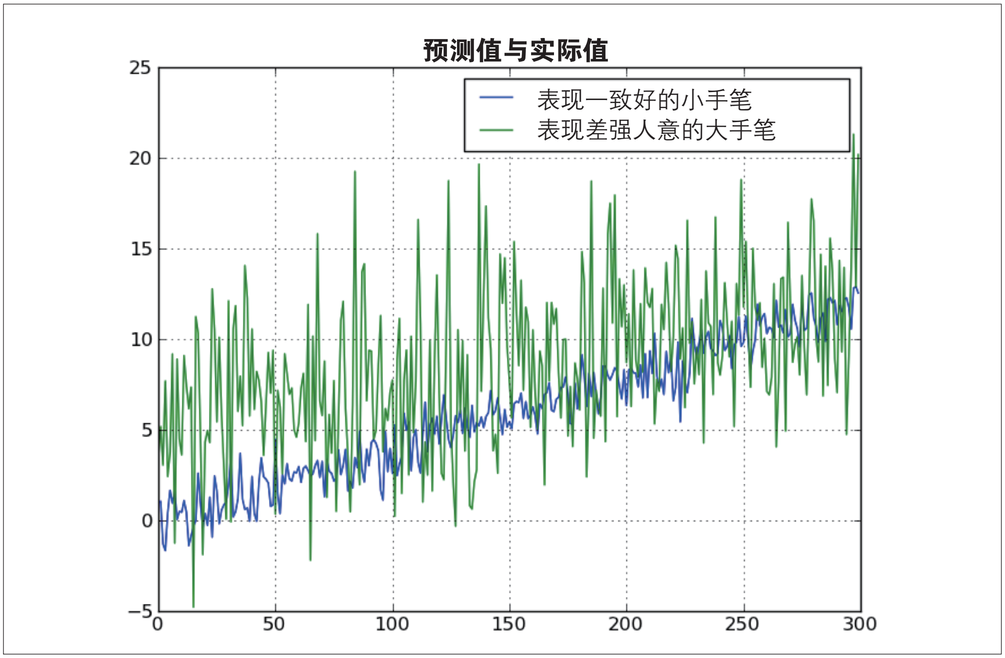


图 6-13：两个理论模型的累积 PnL 值对比图（另见彩插图 6-13）

6.5.8 聊聊回归模型

金融数据中有用的信息含量十分稀少，如果自认为手中掌握的信息可以助你打败市场，还需三思而后行。然而，从形式上倒不妨把金融数据的时间轨迹看作一个连续的、形式未知而复杂的函数。由于函数都可以用泰勒展开表示成一系列成分相加的形式。因此，从形式上，线性模型似乎比较适用于对其建模。

既然线性模型似乎是可行的，那么逻辑回归模型适用与否呢？可以说毫无用处，其对于金融建模的价值基本为 0。因为对于金融数据来说，我们需要的是数据的真实值，越精确越好。逻辑回归的二元输出值，对于理解本来信息量就微乎其微的金融数据来说，无非是火上浇油。

线性模型似乎适用于对金融数据的建模，而且对于“线性”二字的理解不能思维定势。“线性”，从统计学意义上来说，是对参数的线性，也就是，你可以对回归变量取平方或者开方项，甚至是交叉乘积项，只要这样项目最后是加总在一起用来建模，都可以称作线性模型。

6.5.9 先验信息量

先验信息量指在模型建立之前，根据历史、行业经验或者主观判断，对于某些参数的分布做出主观性预判，这种预判被数学化之后并整合进了原先的模型之中。在之前指数平滑的例子

中，我们假设新数据要比旧数据更加重要，因此权重更大。这就是一个直观的先验信息量。

除此之外，我们可能会做出这样的预判："模型参数的变动应该是平滑的。"当我们想利用某些时间序列的已有观测值预测未来值时，这样的预判可能是有用的。比如下面的一个时间序列模型：

$$y = F_t = \alpha_0 + \alpha_1 F_{t-1} + \alpha_2 F_{t-2} + \epsilon$$

其中，对于时点 t 的预测值（y=F_t），模型只用到了前两期的历史观测值。当然，如果认为更多的历史数据可能对预测更加有用，模型也可以引入更多的滞后项。滞后项越多，模型引入的自由度（degrees of freedom）越大，我们可以通过引入更强的先验信息量来抵消这些自由度。简言之，先验信息可以降低模型的自由度。

有关模型参数关系的先验信息量的设定（在这个时间序列模型的例子中就是连续相邻数据点之间的关系）可以通过在回归模型估计时，在原数据协方差矩阵上加一个矩阵的形式完成。欲了解更多信息，请参考：http://goo.gl/gJURp6。

6.5.10　一个小例子

分析时间序列时，一个常用的统计量叫作"自回归系数"，可以帮助我们客观地确定合适的滞后项阶数。比如，图 6-14 就是一个时间序列的自回归系数图，图中的滞后阶数超过了 40，达到了 100 阶。

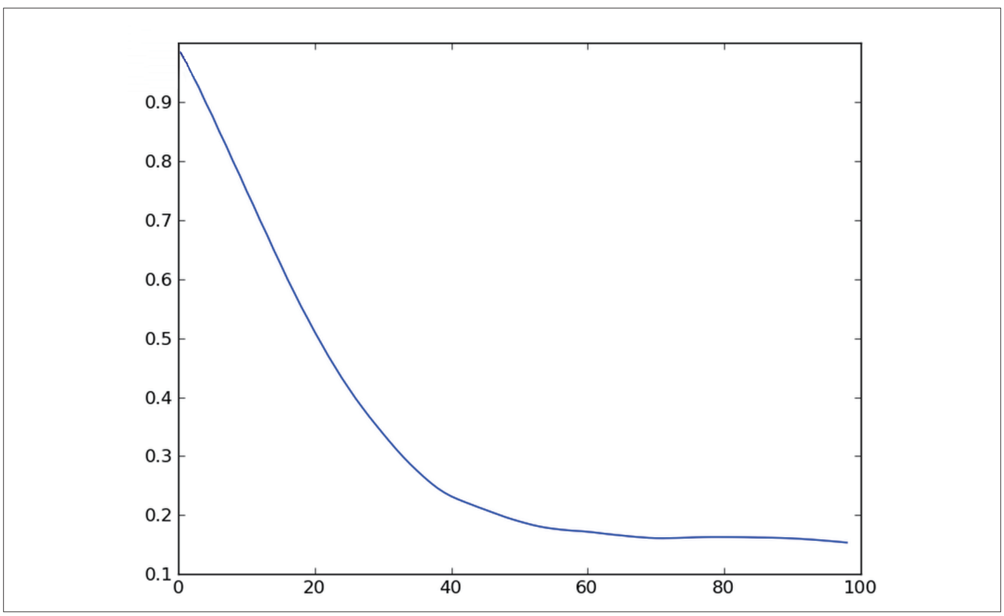


图 6-14：某时间序列的 100 阶自回归系数图

自回归系数的计算很简单。如果要计算 1 阶自回归系数，首先需要将时间序列整体后移一位（其长度要与原序列相同），原序列和后移序列之间的简单相关系数即代表原序列的一阶自回归系数。

从图中可以看出，在 40 阶过后序列的自回归系数已经接近 0.2，这意味着如果想预测下一个值，使用之前的 40 个观测值已经足够。然而，40 个滞后项对于线性回归模型来说还是太多了。要解决这样的问题，之前说到的先验信息量则大有用武之地。

在模型估计的时候，我们经常会通过最小化一个风险函数来得到最优估计值，这个风险函数代表了模型拟合的效果。[7] 从风险函数的角度来理解先验信息是一个不错的角度：结合了先验信息的风险函数也称作“惩罚函数”，举例来说，对于普通的最小二乘回归来说，一个没有任何先验信息的惩罚函数就是离差平方和项：

$$F(\beta) = \sum_i (y_i - x_i\beta)^2 = (y - x\beta)^\tau (y - x\beta)$$

为了得到最小的 F，对 F 取关于 β 的一阶偏导数并令其为 0，则得到 β 的唯一最优解：

$$\beta = (x^\tau x)^{-1} x^\tau y$$

对于离差平方和项 $F(\beta)$，可以加入相应的先验信息量以改善模型的拟合效果。比如说，先验信息表示：模型滞后项中的参数估计的总体规模不能过大，则需要“惩罚”过大的参数估计值。体现在离差平方和中就等同于添加了一个“惩罚项”（$\sum_j \lambda^2 \beta_j^2$）：

$$F_1(\beta) = \frac{1}{\boldsymbol{N}} \sum_i (y_i - x_i\beta)^2 + \sum_j \lambda^2 \beta_j^2 = \frac{1}{\boldsymbol{N}} (y - x\beta)^\tau (y - x\beta) + (\lambda \boldsymbol{I} \beta)^\tau (\lambda \boldsymbol{I} \beta)$$

根据相同的微积分求极值的方法，同样可以得到 β 的最优解为：

$$\beta_1 = (x^\tau x + \boldsymbol{N} \cdot \lambda^2 \boldsymbol{I})^{-1} x^\tau y$$

比较 β_1 和 β 可以看到，添加惩罚项对于模型参数估计的影响等同于给数据的协方差矩阵 $x^\tau x$ 加了一个对角矩阵 $\boldsymbol{N} \cdot \lambda^2 \boldsymbol{I}$。细心的读者会发现，这个对角矩阵其实本质上是单位矩阵的常量积。

很明显，先验信息的形式与模型参数的估计值有着直接联系。让我们再添加一个先验信息，并观察参数的估计会产生什么样的变化。假设“参数之间的变动是平滑的”，也就是说，相邻滞后项的参数估计值之间的差距不应该过大[8]，那么相应的离差平方和应添加一个新项 $\sum_j \mu^2 (\beta_j - \beta_{j+1})^2$：

注 7：比如在回归的例子中，离差平方差就是一个风险函数，通过最小化离差平方和我们可以得到有关模型参数的最优估计。

注 8：对于时间序列模型，这同样是一个合理的先验信息设定。

$$
\begin{aligned}
F_2(\beta) &= \frac{1}{\boldsymbol{N}}\sum_i (y_i - x_i\beta)^2 + \sum_j \lambda^2\beta_j^2 + \sum_j \mu^2(\beta_j - \beta_{j+1})^2 \\
&= \frac{1}{\boldsymbol{N}}(y - x\beta)^\tau(y - x\beta) + \lambda^2\beta^\tau\beta + \mu^2(\boldsymbol{I}\beta - \boldsymbol{M}\beta)^\tau(\boldsymbol{I}\beta - \boldsymbol{M}\beta)
\end{aligned}
$$

矩阵 $\boldsymbol{M}$ 称作移动算子，而 $\boldsymbol{I} - \boldsymbol{M}$ 在离散微积分中称作离散微分算子（参见 http://goo.gl/2D4LeH，以查看有关离散微积分的介绍），该矩阵中除了下三角区域的值为 1 之外，其他区域的值都为 0。$\boldsymbol{M}\beta$ 的作用是将 β 的参数顺序整体向前移动一为，最后的位置用 0 填补，这样的操作对应上式的 $\beta_j - \beta_{j+1}$，这也是为什么 $\boldsymbol{M}$ 称作"移动算子"的原因。

这看起来相当复杂，因此在计算的时候就需要多加小心。$F_2(\beta)$ 对于 β 的微分要比之前的复杂很多，这里涉及向量微积分的内容，需要知道：

$$
\frac{\partial\mu^\tau\cdot\mu}{\partial\beta} = 2\frac{\partial\mu^\tau}{\partial\beta}\mu
$$

再求 $F_2(\beta)$ 对于 β 的偏导数：

$$
\begin{aligned}
\frac{\partial F_2(\beta)}{\partial\beta} &= \frac{1}{\boldsymbol{N}}\frac{\partial(y - x\beta)^\tau(y - x\beta)}{\partial\beta} + \lambda^2\cdot\frac{\partial\beta^\tau\beta}{\partial\beta} + \mu^2\cdot\frac{\partial((\boldsymbol{I} - \boldsymbol{M})\beta)^\tau((\boldsymbol{I} - \boldsymbol{M})\beta)}{\partial\beta} \\
&= \frac{-2}{\boldsymbol{N}}x^\tau(y - x\beta) + 2\lambda^2\cdot\beta + 2\mu^2(\boldsymbol{I} - \boldsymbol{M})^\tau(\boldsymbol{I} - \boldsymbol{M})\beta
\end{aligned}
$$

令其为 0，则可以得到最优解：

$$
\beta_2 = (x^\tau x + \boldsymbol{N}\cdot\lambda^2\boldsymbol{I} + \boldsymbol{N}\cdot\mu^2\cdot(\boldsymbol{I} - \boldsymbol{M})^\tau(\boldsymbol{I} - \boldsymbol{M}))^{-1}x^\tau y
$$

与 β_1 的区别之处就在于我们在协方差矩阵 $x^\tau x$ 上又添加了新的项 $\boldsymbol{N}\cdot\mu^2\cdot(\boldsymbol{I}-\boldsymbol{M})^\tau(\boldsymbol{I}-\boldsymbol{M})$，该项代表了之前设定的"参数变动应该是平滑的"先验信息对最终参数估计的影响。注意，对称矩阵 $(\boldsymbol{I}-\boldsymbol{M})^\tau(\boldsymbol{I}-\boldsymbol{M})$ 的上下两条次对角线上的元素都为 1，而主对角线上的元素为 2。也就是说，当我们在调整 μ 时 λ 也应该做相应调整，因为它们俩之间具有联合变动关系。

先验信息与高阶导数

刚才先验信息的设定只涉及了一阶导数。实际上，你也可以在二阶导数或者更高阶导数上设定先验信息。如果在二阶倒数上设定先验，则相应的 $(\boldsymbol{I}-\boldsymbol{M})$ 变为 $(\boldsymbol{I}-\boldsymbol{M})^2$。以此类推，高阶导的先验信息对应 $(\boldsymbol{I}-\boldsymbol{M})$ 的高阶幂值。

那么，到现在为止，我们的模型到底是什么呢？从参数来看，我们需要估计平滑指数 γ，另外还需要估计 λ 和 μ：分别对应刚才解释过的 $x^\tau x$ 和 $x^\tau y$ 项的移动估计。因此，最后的模型涉及三个需要估计的参数：γ、λ 和 μ。通常来说，对于需要多大的指数平滑值（γ）我们心里大致有数，但是对于 λ 和 μ 的选择则取决于数据本身，以及它们之间的相互影响。对这些模型参数的最优化估计无非是一件颇需工匠精神的事情：既要保证这些参数的值得到了恰当的优化，又要避免由于过度优化可能导致的模型过拟合问题。

6.6 练习：GetGlue提供的时间戳数据

GetGlue 公司为我们提供了一个时间戳数据集以供练习。这些数据与用包含用户签入并评论电视和电影节目的时间戳事件。

原数据（http://bit.ly/1aL8XS0）的时间跨度从 2007 年到 2012 年，其中每个用户对应一份数据。数据量占到 GetGlue 公司用户数据库总量的 3%，但即便如此，解压缩之后的数据也有 11 GB。

运行下面的 R 代码可以看到该数据库的前 10 条信息：

```
#
# 作者：Jared Lander
#
require(rjson)
require(plyr)

# 导入数据
dataPath <- "http://getglue-data.s3.amazonaws.com/
             getglue_sample.tar.gz"

# 建立连接以解压缩得到的文件
theCon <- gzcon(url(dataPath))

# 读取前 10 行数据信息
n.rows <- 10
theLines <- readLines(theCon, n=n.rows)

# 检查数据结构
str(theLines)
# 注意，数据向量的第一个元素与其他行有所不同
theLines[1]

# 除了第一个元素，应用 fromJSON 到向量剩下的所有元素
theRead <- lapply(theLines[-1], fromJSON)

# 转换成数据框的形式
theData <- ldply(theRead, as.data.frame)

# 检查刚才所有操作的最终效果
View(theData)
```

我们给出以下的分析步骤，仅供参考。

(1) 先读取 1000 行的数据并仔细逐条分析。在对这 1000 个数据有了充分的了解之后，可以考虑读进 10 万行数据或者上百万行数据做更深入地分析。
(2) 做探索性分析，尤其是能够汇总并尝试回答以下几个问题。
  - 用户的可能行为有多少种？用户行为的大概分布如何？

- 数据中有多少个不同的用户？[9]
- 哪 10 部电影最受欢迎？
- 在 2011 年有多少部电视和电影节目？

(3) 根据探索的结果，尝试问自己 5 个新问题。
(4) 尝试回答自己的 5 个新问题。
(5) 可视化！如果觉得数据中的某个特征比较有意思，尝试将它们画出来以更好地展示给自己或者别人看。

练习：金融建模

关于金融建模方面，尝试完成以下任务。

(1) 获取数据：从雅虎网站的财经板块下载一只至少有 8 年观测长度的股票每日价格和成交量数据。这是最起码的一步，如果你不知道怎么做，可以到网上搜索一下。
(2) 计算该股票的日度对数收益率。
(3) 用同样的方法计算对数成交量的变动值，并与收益率进行比较。
(4) 对收益率序列和成交量序列分别建立线性回归模型，并使用至二阶滞后项。用预测值与真实值对比，如果预测值十分准确，没准你能从中赚一笔。如果你还能尝试建立因果模型，对数据进行标准化操作，或者在模型中使用指数平滑，你已经迈入高手行列了。
(5) 画出模型的累积 PnL 图，并与书中的图 6-13 比较。

注 9：需根据用户的 ID 信息分辨重复用户。

第 7 章

从数据到结论

想知道一些公司是如何从数据中提取信息的吗？

本章将由 Kaggle 公司的 William Cukierski 以及谷歌的 David Huffaker 为大家现身说法。

7.1 William Cukierski

William 在康奈尔大学获得物理学学士学位，博士毕业于罗格斯大学，研究领域是生物医学工程，研究方向主攻癌症的病理图像研究。在撰写博士论文间隙，William 参加了 Kaggle 上的一些数据分析竞赛，并取得过骄人战绩。现在，他在为 Kaggle 工作。（下文会有关于 Kaggle 以及 Kaggle 竞赛的详细介绍。）

William 首先会为大家介绍一些关于数据科学竞赛和数据分析众包业务的内容。另外，他还会详细介绍有关 Kaggle 竞赛平台的细节。最后的重头戏是关于特征提纯和特征选择的内容。特征提纯是将原始数据中的垃圾信息或者变量剔除，让数据更加“干净”。对于数据分析来说，数据的质量会直接影响模型的预测效果。特征选择是在数据提纯之后，根据数据和研究问题的特点，通过提取、重构、组合等方式构造对于建模有用的预测量。

7.1.1 背景介绍：数据科学竞赛

数据科学竞赛是机器学习领域高手们切磋武艺的常见形式，从其出现至今已经有一段历史了。一般来说，主办方会设计一个较为前沿的机器学习任务，并免费提供分析用的数据。参赛人员 / 队伍被要求在短时间内（通常是几周或者几个月）设计出相应的机器学习预测模型。至于竞赛的任务则多种多样，有些竞赛要求设计算法，预测人们遭遇车祸的可能

性，或者人们喜欢某部电影的可能性等。除了免费提供数据、制定比赛规则和注意事项以外，主办方会公布统一的模型评价标准以及参赛者提交模型的规则等。

比较著名的数据科学竞赛有一年一度的 KDD 竞赛（知识发现和数据挖掘竞赛），Netflix 公司的 Netflix Prize 竞赛（奖金高达 100 万美元，前后持续了两年），以及我们待会要详细介绍的 Kaggle 竞赛。

关于数据科学竞赛有必要在这里做三点声明。首先，作为数据科学生态系统的重要组成部分，数据科学竞赛在一定程度上推动了数据科学的发展。作为数据科学家，应该密切关注数据科学竞赛的最新动态。

其次，数据科学竞赛的内容和形式在一定程度上定义了数据科学本身。竞赛的不断发展给数据科学的应用建立了很好的内容库，同时也在不断地更新着数据科学的定义。当然，这并不代表数据科学竞赛的内容直接定义了数据科学的内容，只是说，在一定程度上，竞赛的内容为了解和推广数据科学本身提供了鲜活的材料：数据科学到底是在处理什么样的问题，数据到底长什么样子，数据科学忽视了哪些因素等。

第三点，数据科学竞赛的结果往往都会以个人或者团队排名的形式公布，排名靠前的往往被视为顶尖的数据科学家。然而事实上，很多优秀的数据科学家，尤其是女性，并没有参加任何形式的数据科学竞赛。这代表她们不可能获得任何排名，但她们却同样优秀。因此，我们应该客观理性地看待竞赛的排名。

数据科学竞赛：到底谁在参与竞赛

各种各样的数据科学竞赛看来十分正规、程式化并且有条不紊。主办方已经准备好了数据、模型的评价标准以及需要解决的问题，甚至对于建模之后如何可视化和报告都有详细的指导。比如说 Kaggle 公司的数据科学家就是在做这些事情：找有意义的数据，做好基本的清理工作，尝试建模，寻找合理的评价标准，构思有意义的分析问题等。这些工作本身应该成为数据科学的一部分，应该成为数据科学家必备的技能。而主办方似乎把这些事都办得妥妥的，数据科学竞赛中“数据科学”的味道已经被大大地弱化了。

7.1.2　背景介绍：众包模式

这里讨论以下“众包模式”的概念，总体来说有两种众包模型。第一种叫作分布式众包模式，其典型代表是维基百科。这种众包模式适合任务相对简单而贡献来源分布较广的情形。在维基百科上，来自世界各个角落的人都可以贡献内容，由志愿者组成的幕后团队负责管理和控制内容的质量。维基百科制定了一套完整的内容质量管理系统。这种众包模式的结果是集众人的知识和力量搭建了一个内容可信度极高的在线百科全书。

另外一种众包模型解决的是更加专业、困难和具体的问题。Kaggle、DARPA、InnoCentive等很多公司都在从事这样的众包业务。这些公司将某个较为专业的问题公布于众，往往只有很少的一部分人具备解决该问题的能力，因此竞争只在这一小群人之中展开。获胜者往往可以得到可观的现金回报，当然还有随之而来的荣誉和业界的肯定。

后者这样的众包模型，其最大的短板是是参与者数量太少。究其原因，首先很多竞赛往往没有或者缺乏合理公正的模型评价标准。在一些竞赛中，对于输赢的评价标准往往不够客观，掺杂着个人主观因素。比如说，你的模型是好是坏单单由评委说了算，而评委的喜好也各不相同。其次，参赛者的沉没成本相当高，因为只有获得名次才有奖励和荣誉，只有金字塔顶的人才能拔得头筹，往往还需要一定的运气。这些都直接导致了数据科学竞赛的参与者寥寥。

组织问题也同样妨碍了数据科学竞赛的健康发展。有些随意设计、组织混乱的竞赛往往让参赛者分析一些毫无意义、枯燥无比的数据。这些组织者往往认为无论出什么题、给多少奖金，参赛者都会摩拳擦掌，跃跃欲试。这些无意义的竞赛严重打消了数据科学家们的积极性。有些竞赛的题目要么过于庞大，让人找不着北，要么过于细碎，让人兴致全无。

既然已经认识到了这么多妨碍众包模式发展的因素，我们可以预期一个好的众包竞赛首先应该精心设计竞赛题目，做到有趣、可行又有实际的商业意义。模型的评价标准应该足够透明和客观。为了充分提起参赛者的兴趣，奖金当然越高越好，不同名次奖金的分配和颁发，也应该合理而透明。

众包竞赛的发展其实已经有了几百年的历史，下面是一些影响力较大的竞赛，让我们回顾一下。

- 公元 1714 年，英国皇家海军因为地球经度的测量问题[1]伤透了脑筋，于是拿出相当于现在 600 万美元的奖金征求能够解决该问题的人。一位名不经传的家具工 John Harrison 巧妙地利用钟表原理制造了一种叫作“经线仪”的工具解决了这个问题。
- 2002 年，福克斯电视网络公司推出大型电视选秀节目“美国偶像”，意在海选出下一代流行歌王 / 歌后。该节目开创了电视选秀节目的先河，选手在一轮又一轮的淘汰赛中比试歌喉，最终获胜者可得到巨额奖金。
- X-prize 公司（http://www.xprize.org/）是一家专注于大型科学竞赛的推广公司。这些竞赛的内容往往涉及与人类生存息息相关的重大科学问题，一旦解决，可以为传统产业注入新生血液，甚至会开创一个全新的产业。因此，竞赛的另一大特色就是：奖金奇高。其中的 Ansairi X-prize 是一个空间科学竞赛，其奖金高达 1000 万美元。这样的问题往往需要科学界的顶级专家，花费大量的时间才能解决。因此，这样的众包竞赛由于门槛过高，参与人数少，往往要持续很长的时间。但是，这对于相关高精尖领域科学的发展还是明显有推波助澜的作用。

注 1：如何确定船舶在东西方向的具体位置。

注 2：当然，也可能没表情。

关于众包和土耳其机器人

众包和土耳其机器人这两个名词在数据科学领域悄然地出现，并且越来越多地引起人们的关注和重视。

虽然“众包”一词的正式出现是在2006年，但其概念本身却并不新鲜：针对一个问题，多人一起各自独立提出解决方案，综合一起就会得到一个更佳的方案。James Suriowiecki 的 *The Wisdom of Crowds*（《群体的智慧》）（Anchor，2004）一书详细讲述了众包的理论含义，其核心观点是：三个臭皮匠顶过一个诸葛亮。当然，群体智慧发挥作用需要一定的条件，其中首要的是“独立性”，也就是群体中每个个体应该独立思考并提出自己的解决方案，而不是采取群体讨论的方式给出小组答案。当然，并不是所有的问题都适合交给三个“臭皮匠”来解决，有些问题还是需要“诸葛亮”（领域专家）的帮助。

亚马逊推出的“土耳其机器人”业务是线上众包服务的代表。比如，表情识别对于人眼来说是个很简单的问题，看一眼就知道表情是“开心”还是“难过”[2]，但这对于机器学习来说是个极富挑战性的问题。对于表情识别来说，监督性学习的目的就是给定一批图像的标签（是“开心”或“难过”），通过算法学习和识别新图像中的表情是“开心”还是“难过”。问题是，如何事先得到一批已经做好标签的图像呢？这就是“土耳其机器人”业务的出发点，因为这样的贴标签任务只能通过人工手动完成，“土耳其机器人”业务就是为用户提供图像贴签服务。网络上的每一个人都可以注册成为亚马逊的“土耳其机器人”，只要你有闲暇时间就可以选择坐下来给一些图像贴标签，正好还可以赚点外快。因为这样的任务没有太多技术含量，也比较枯燥乏味，固冠名为“机器人”。当然，为了防止“机器人”们闭着眼睛瞎填，亚马逊设计了严格的质量控制系统。一旦发现作弊，你就永远失去了做“机器人”赚外快的资格。

“土耳其机器人”看似是人的体力劳动，但其终极目标是结合机器学习算法和少数人的劳动成果，最终减少了大多数人的重复性劳动量。因为那些被贴好标签的数据会被各种各样的监督性机器学习算法用来预测大量的未被贴标签的数据。

7.2 Kaggle模式

数据科学家走在通往无所不知的大路上，走到尽头才发现，自己一无所知。

——Will Cukierski

Kaggle 的口号是“让数据科学成为一项竞技赛事”。Kaggle 是其他公司与数据科学家之间的联络人。比如说，一家公司想将数据分析任务众包给数据科学家，Kaggle 会揽下该活并收取一笔佣金。因为 Kaggle 提供的数据科学竞赛平台吸引了来自世界各个角落的数据科学高手，这些众包的任务只需要交给自家平台上的数据科学家即可。

当然，Kaggle 公司内部也是藏龙卧虎之地，许多顶尖的数据科学家都在 Kaggle 工作，Will 就是其中的一员。提供分析任务和数据，并付钱给 Kaggle 的是需要数据分析众包的公司，而解决这些任务的是 Kaggle 平台吸引的来自世界各地的数据科学家。Kaggle 的数据科学平台，任何人都可以注册参赛，基本没有门槛限制。下面我们会分别从参赛者和需求方公司的角度分析一下 Kaggle 的业务模式。

7.2.1 Kaggle的参赛者

在 Kaggle 的数据竞赛中，参赛者可以拿到一个训练数据集和一个测试数据集[3]。竞赛的题目大多数是有关监督性机器学习的，也就是说在训练数据集中，x 和 y 的值都是给定的，用来训练监督性学习算法。测试数据集中只会提供 x，不提供 y。参赛者在训练数据集上训练模型，并将模型应用于测试数据集，得到预测的 y 值。如果参赛者对模型的效果满意，可以在 Kaggle 平台上提交预测的 y 值。Kaggle 的后台系统会根据事先设定好的模型评价标准，比较参赛者提供的 y 值和真实 y 值的差距，并给出参赛者的实际得分值。在这个过程当中，参赛者不需要上传代码而只需要上传模型的预测值，因此 Kaggle 并不在意你用什么分析软件。当然，最终的获胜者需要提供代码。测试数据集中的 x 值虽然不包含真实的 y 值信息，却也十分重要，它无论在变量个数、含义和格式方面都和训练数据集中的 x 值完全一样。为了保证比赛绝对公正，尽量降低作弊的可能性，在竞赛截止日期之后，Kaggle 会用第三个数据集[4]测试参赛者最终的模型效果，并据此判定最终的排名。这个数据集无论是 x 值还是 y 值，参赛者都无从得知，从而可以最大程度上保证数据的独立性和模型效果的客观性。[5]

为了防止参赛者频繁地提交，Kaggle 的竞赛一般限制参赛者一天最多只能提交 5 次，每个竞赛持续几周到几个月不等，甚至更长。每次提交，Kaggle 可以迅速计算出模型的效果并通过排行榜的形式反馈给参赛者。排行榜上可以看到每个参赛者的名字和最后提交的模型的预测误差。如果一个赛事有足够多的参赛者，可以看到图 7-1 中那样“前赴后继”的壮观场景。只要一个参赛者的模型有了细微的改进，别的队伍会迅速跟进。比赛的效果很明显，每个参赛者的模型效果在比赛期间都有或多或少的改进，最后的冠军凭借 0.791 的模型准确度拔得头筹。可以想象，如果没有这样一个竞赛平台而只是单干的话，数据科学家不断改进模型的本能和冲动很难被激发出来。

注 3：需要同意一个数据使用协议。

注 4：不同于参赛者拿到的数据集，也不是竞赛期间 Kaggle 后台评分用的数据集。

注 5：Kaggle 在竞赛期间会设立排行榜，在排行榜上的获胜者并不是最终的获胜者。因为排行榜的结果来自于提供给参赛者的测试数据集，最后的独立数据测试结果决定了最终的冠军归属。

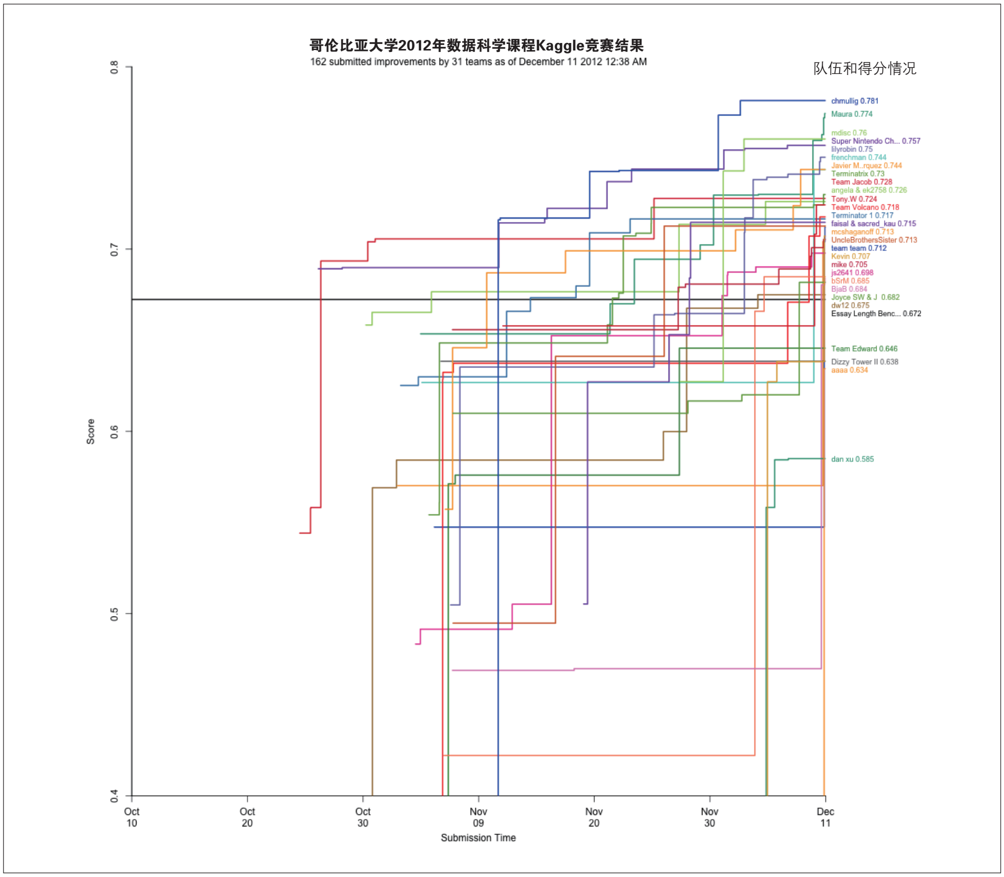


图 7-1：该图出自 Chris Mulligan，他是 Rachel 班里的学生。该图很好地描述了每个参赛个人 / 队伍在比赛期间，模型的进化情况（另见彩插图 7-1）

这样“前赴后继”改进模型的场景固然会带来预测效果更好的模型，但同样也会带来更加复杂的模型。为了不断改进模型，参赛者会使出浑身解数。这也是为什么一般的竞赛至多只会持续数周或者数月的时间，因为模型的预测精度总有上限，在进化一段时间之后就会进入瓶颈期，要想提高模型的精度，参赛者会把模型打造得越来越复杂。以 Netflix Prize 为例，这个为期两年的竞赛，其最后的获胜模型由于过于复杂，Netflix 公司都没有在实际产品开发中应用该模型。[6]

7.2.2　Kaggle的客户

Kaggle 的客户是一些需要数据分析的公司，那么这些公司为什么愿意付钱给 Kaggle 呢？原因很简单，因为 Kaggle 的平台上有许多活跃度很高的数据科学家们。一些公司存储了海

注 6：当然，奖金还是如数发给了获胜队伍。

量的数据，并且每天都有大量的新数据产生，但苦于没有能够分析这些数据的科学家们，Kaggle 作为中间人恰好填补了这个空白。个别公司会自己设计数据竞赛吸引数据科学家们。Kaggle 模式的创新之处就在于，它凭借自身的平台优势，说服客户捧出他们手中宝贵的数据。客户往往会欣然接受，因为 Kaggle 平台上的数据科学家们总会帮助解决令他们头疼的数据分析问题。

迄今为止，Kaggle 竞赛已经为业界带来了不少优秀的数据分析解决方案。Allstate 是一家汽车保险公司，虽然他们公司内部已经有一只素质良好的精算师团队，他们还是把数据放在了 Kaggle 上寻求更优秀的解决方案。Allstate 给出的分析任务是：给定汽车驾驶员的属性信息，预测其发生车祸的可能性。Kaggle 上一共有 202 个数据科学家参与了这项竞赛，竞赛的结果令人瞠目结舌。冠军的模型效果，以正则化的 Gini 系数为标准，将原本 Allstate 公司的模型效果提升了 271%（详情可见 http://www.kaggle.com/solutions/casestudies/allstate）。这样的例子还有很多，其中某公司在 Kaggle 上发布了奖金额仅为 1000 美元的竞赛项目，最后从中的获益，据保守估计，超过数十万美元。

这样公平吗？

与前面的 Allstate 例子类似，这些公司内部已经有了一支素质良好的数据分析团队，如果这些公司还在 Kaggle 上寻求解决方案，对公司内部的数据科学家固然不太公平。如果 Kaggle 竞赛的获胜模型明显优司内部开发的模型，这些员工很可能会被扫地出门。

竞赛中只有排名前三（甚至第一名）才有可能拿到奖金，这意味着绝大多数人都在免费地拼命干活，而公司往往会从 Kaggle 的获胜模型中获益颇丰，这对绝大多数参赛的数据科学家们公平吗？

Kaggle 会对这些公司收取佣金，有些公司所公布的奖金额也十分慷慨，至于参不参赛，数据科学家有着绝对的自主权。

这些因素加起来，Kaggle 对于三方来说似乎都是公平的。真是这样吗？

其实得益较多的应该是 Kaggle 的客户，数据科学家往往意识不到他们所开发的模型以及花去的时间价值要远远大于奖金额。除非对某个竞赛题目或者数据由衷得感兴趣，否则在从事这个基本上等同于零回报的数据竞赛之前，数据科学家们还是要三思而后行。

最近 Facebook 在 Kaggle 上发布了一项数据科学竞赛，获胜者可以获得 Facebook 的面试机会。最后的参赛人数达到了 422 人。我们认为这极大地方便了 Facebook 招募优秀的数据科学家，但是 Cathy 却觉得这很可能会把数据工作者们引入歧途：他们专注于分析数据和解决问题，却忽视了公司政策和文化底蕴，这对于公司和员工来说同样重要。

Kaggle 的作业自动评分模型

还记得本书开头提到的，在哥伦比亚大学开设的数据科学课吗？作为该课期末考核的一部分，学生被要求建立一个作业自动评分系统。考核方式完全参考了 Kaggle 竞赛的模式，学生之间也可以组队完成任务。下文将详细介绍该作业自动评分系统的内容，数据可以从 https://inclass.kaggle.com 获得。

训练数据集是一批已经被老师批改完的作业，*y* 值是作业的最终得分，*x* 值是作业的相应特征变量。测试集只有 *x* 值，会被用来评判模型的最终预测效果。

竞赛中一共用到了 5 组不同的作业数据集，每个作业数据集都生成自一个单独的提示语（prompt）。每篇作业的总字数在 150 到 550 之间。某些作业可能跟原信息源相关，而有些则与信息源无关。完成每篇作业的学生其成绩都在 7 到 10 之间。每篇作业都由两名老师独立手工批改并打分。除了这些共同特征变量，每组作业还有相应的特色变量。数据所特意生成的特异性旨在测试评分系统引擎的灵活程度。具体来说，数据集会包含以下几列：

- id
  作业的独立编号
- 1~5
  作业来自的组号。一共有 5 组不同的作业。
- essay
  用 ASCII 码表示的作业的具体内容
- rater1
  第一个老师的打分
- rater2
  第二个老师的得分
- grade
  综合分数

7.3 思维实验：关于作业自动评分系统

我们在课堂上问过学生们对作业自动评分系统持什么态度，他们是否愿意该系统去批改他们写的作业，以及这个系统可能会带来什么样的好处和坏处，等等。下面是一些学生的回答。

- 自动评分系统更加客观。
  已经有研究表明，医生在两个月前后对同一张 X 光片的诊断会截然不同。这很正常，因为人的思维具有不一致性。即便你认为自己毫无偏颇，然而事实就是事实。从这一点上来看，机器给出的判断往往比人的判断要更加客观，前后一致。机器学习甚至已经被广泛应用在了癌症研究上，要知道这是一项高风险的任务，但是机器似乎并不比人做得差。

- 机器过于程式化，因此缺乏创新能力？

  机器评分可以使整个批改任务标准化、流水线化。人们通常喜欢标准化的东西，这样虽然保守，但有更好的一致性。比如说汽车，相比于手工打造的汽车，人们会更加信任流水线生产的汽车，因为它更加安全可靠。但问题是，并非所有的任务都适合程式化。至于批改作业到底适不适合标准化，这还是要考量作业的内容、形式、难度等诸多因素。

- 考试的目的就只是要学生们写出好论文吗？

  这里假设考试的形式是一篇小型论文，通常老师会给出详细的写作大纲和评分标准。如果严格地按照大纲和标准来说，得高分并不是一件难事。那些培训机构甚至专门著书教导学生们如何应试。那么这种应试技巧有没有办法翻译成机器语言呢？一个可能的办法就是设计机器学习算法，通过老师给出的大纲和标准，自动化生成论文。如果真是这样，教育过程就变成了机器之间的竞技——是学生和老师的算法之间进行博弈。而在这场战役中，我们认为学生的算法获胜可能性更大。

领域知识与机器学习算法

领域知识和机器学习并非水火不容的关系。恰恰相反，数据科学问题的解决离不开二者中的任何一个。然而 Kaggle 的主席 Jeremy Howard 先生在 2012 年 12 月份的《新科学家》杂志接受了 Peter Aldhous 的专访时说道：“专家的知识基本上毫无用处，我们根本不需要。”这下可把专家们惹毛了。下面是那次专访的实录，我们来听听他到底说了些什么。

PA：Kaggle 比赛的获胜者和表现平平者有何不同？

JH：从技术上来看，Kaggle 比赛获胜的关键是将最有用的信息交给模型。因此，你必须决定从原始数据中抽取多少信息、何种信息、以什么样的方式交给模型。Kaggle 比赛的获胜者一般都具有强烈的好奇心和创新能力，对于同样的问题，他们擅于从许多不同的角度解析、分拆和组合问题。一些像随机森林这样的算法并不在乎你有多少想法，只要你的想象力足够丰富，只管喂给模型就好了。随机森林会自动使用那些更加有用的信息。

PA：这听起来与传统预测模型的建立过程完全不同，通常领域专家会扮演十分重要的角色，决定采用或者不采用哪些特征变量。这些专家对 Kaggle 有何评论？

JH：真要我说的话，可能会得罪很多人。我想说的是，这些专家积累几十年的所谓“领域知识”基本上毫无用处，我们在建模时根本不需要领域专家的帮助。专家们所使用的看似花哨的模型，其效果比单纯的机器学习算法要差很多。当然，这个观点很多人会驳斥，因为在过去几十年内，专家的意见总是主导着人群的主流意见。但是他们总是花很长的时间讨论一个变量是否有用，在细枝末节上钻牛角尖，我真的认为是在浪费时间。

PA：那你觉得专家的知识还能在哪派上用场？

JH：也许在建模初期会有用，因为专家们通常比较擅于发问，他们可以帮助我们提出更好的问题。但接下来就是数据科学家的战场，基本跟他们没有什么关系了。

PA：Kaggle 上充斥着大量的数据驱动的、黑匣子似的算法，即便是建模者自己也不知道模型到底如何解释。你觉得这种方法是否存在弊端？

JH：有些人认为，使用算法解决问题的弊端在于，即使最后得到了答案，也不代表着你对问题有更深入的认识。但模型的解释性果真如此重要吗？我觉得不然。算法会告诉你哪些变量重要，哪些变量不重要，这已经足够了。你或许会问，这些变量为什么重要，而为什么这些变量不重要。我觉得这些问题就很无趣了：既然你已经得到了一个预测效果相当好的模型，就不必再对模型的可解释性吹毛求疵了，这完全没有什么必要。

7.4 特征选择

在数据建模中，数据科学家从数据中选取哪些特征变量，以怎样的形式加入模型，这个过程叫作特征选择。

特征选择对模型的效果往往有着直接影响。Will 在被 Kaggle 招安之前，屡次在各个 Kaggle 数据科学竞赛中斩获佳绩（这也是为什么 Kaggle 会招他的原因），因此他对怎样建立一个有效的模型有着充分的发言权。特征选择不仅可以帮助你在竞赛中取得好成绩，更重要的是，它直接的影响模型或者算法的效力。原始数据固然隐藏了所有的信息，但只有通过特征选择攫取出来的信息才能为模型所用。

原始数据中存在大量的冗余信息，比如很多相关性很大的变量。如果一股脑得全放进模型，必然不会有很好的效果。这些信息需要经过挑选、转化或者组合，才能为模型所用。比如说，对变量取对数，将连续性变量离散化为二元变量都是基本的转化形式，它们往往会在一定程度上提高模型的预测能力。

术语解释：特征、解释变量和预测变量

不同的学术领域会用不同的术语表述相同的对象。统计学家喜欢用“解释变量”“因变量”或者“预测变量”等表示模型的输入变量。而计算科学家往往统称为“特征”。

> 特征提纯和选择对于机器学习的重要性经常被人忽视，但却是其中最为重要的一环。选择更好的特征变量往往可以得到效果更佳的模型。
>
> ——Will Cukierski

我们使用的算法与人无异，我们只是数据比他们多而已。

——Peter Novig，谷歌研究总监

Will 觉得 Peter Novig 所提到的更多的数据，应该指的是*更好的特征*。因为更多的数据有时候并不会带来额外的信息。（比如说，你关心掷骰子过程中 2 出现的概率，那么掷骰 1000 次骰子和 10 000 次骰子所得到的结果差别并不大，因为很可能当你掷第 500 次的时候，2 出现的概率已经稳定在了 1/6 左右，并且不再有明显的改变。因此更多的数据在这里并不代表更多的信息。）谷歌公司所要解决的数据问题往往都纷繁复杂，因此搜集更多的数据往往能够带来更多更有用的特征变量，这也许是 Peter Novig 真正想表达的意思。

然而，过多的特征也并不见得就绝对是件好事。比如说，当数据中特征变量的个数超过观测值的个数时，会产生稀疏性的问题。这会导致很多模型的参数估计失效。因此，过多的变量也并非是好事。有时候，太大的数据还会带来额外的存储问题和计算问题：因为数据太大，一个计算机的内存很难一下子装进如此多的数据。因此数据必须被分割，存储在不同的计算机上。这是数据科学中一个令人头疼的问题，它往往会带来不小的负面作用。

然而，总体来说，要想得到好的预测模型，特征选择是不可逾越的一步。

7.4.1　例子：留住用户

假设要你设计了一款叫作“追龙”（Chasing Dragons）的手机游戏（程序图标如图 7-2 所示），用户以交月费的形式使用该程序。那么很明显，用户数量越多，你赚的钱就越多。然而事实上，只有 10% 的新用户会在下一个月度选择续订。因此如何吸引更多的新用户，以及如何留住老用户是你需要考虑的两个最重要的问题。通常来看，招揽新用户的成本要大于留住老用户的成本。但是我们这里只关心其中一个问题，就是如何更有效地留住用户。应用机器学习模型，我们可以根据新用户的特征*预测*该用户下一个月是否会继续续订你开发的这款游戏。模型本身也会有助于你*理解*产品特征、用户行为等对用户黏性的影响。但我们更加关心是模型的*预测*效果到底好不好。如果模型的预测效果很好，对于那些可能会退订的用户，你可以采取相应的激励措施（比如一个月的免费使用权）想方设法留住该用户。

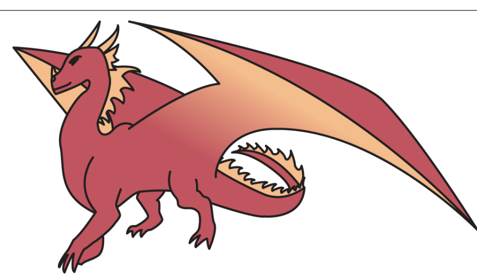


图 7-2：你设计的手机游戏程序：“追龙”

什么模型比较适合呢？第 4 章中，我们说到了逻辑回归模型主要适用于预测二元变量，因此可以先试试逻辑回归模型。模型的输出值是用户在下一个月续订的概率值。（统计学中的生存模型在这里可能更加适用。）在用户下载程序的那一刻起，数据的收集工作就开始了。在第一个月的 30 天时间里，要尽可能详细地记录每位用户的使用行为习惯等，并根据用户的行为数据预测其在下一个月续订的概率值。数据的形式可以是时间戳的形式，比如：用户 A 在早晨 5：22 分打开了游戏程序，5：23 分杀死了一只龙，在 5：24 分拿到了 24 分，5：25 分游戏自动生成了一条有关香体露的广告，等等。严格来说，用户的每一个动作都可以也应该被详细地记录下来。

活跃用户和非活跃用户之间的行为差别巨大，前者会产生大量的行为数据，而后者可能只有零星的记录。在数据分析前，数据需要被转化为可分析的矩阵形式，行代表用户，列代表用户的行为特征变量。在数据收集和整理阶段，原则上应该完整记录下用户的每个行为特征，不做任何删减，这个阶段称作"特征生成"，而不是"特征选择"。但是，应该构建哪些行为特征变量可是需要团队紧密协作的，团队成员应该坐下来进行激烈的头脑风暴。因为这对之后的模型效果有着直接的影响。下面是一些可能的变量，可作参考：

- 第一个月内用户访问游戏程序的总次数；
- 用户第一次使用游戏程序的持续时间；
- 第一个月内，用户每天的游戏得分（也可以是 30 个变量，每个变量代表一天的得分）；
- 第一个月的游戏总得分；
- 用户是否填写了完整的个人信息（二元变量，1 代表已填写，0 代表未填写）；
- 用户年龄；
- 用户性别；
- 用户手机的屏幕尺寸。

这样的特征变量越多越好，即便有些变量之间的差异可能很小，或者相关性很强，也没有关系。

特征生成与特征提取

刚才所说的"特征生成"过程也称作"特征提取"。这个过程既是一门科学，又是一门艺术。做特征提取时，有领域专家从旁指导固然是好，若没有的话，充分发挥你的想象力也会收到不错的效果。

只要想象力足够丰富，你可以得到成千上万的特征变量。如此多的特征变量对于像问卷调查这样的学科是难以想象和企及的。有经验的人会知道，一篇问卷的问题一般不过几十个，受访者能够仔细真实回答的也不过 20 个。

如果特征变量过多，难免会有一些是不太有用的。有些你能够想到的变量，在数据里也并不一定能够找到。因此，在头脑风暴这些特征变量的时候，可能会遇到以下几种情形。

- **有些变量可能与问题十分相关，对建模也十分有用，但是在数据中并不一定存在。**
  用户的很多信息是我们无法记录的，比方说，用户的闲暇时间；用户在玩哪些别的游戏，使用哪些别的程序；该用户是否失业下岗；该用户是否失眠；用户是否容易对游戏上瘾；用户是否对龙比较反感，以至于会做恶梦。这样变量固然对我们理解用户的行为，以及后期的建模都十分有益，但问题是这些数据大多是很难直接获取的。当然，有些变量我们可以通过间接的方式获得。比方说，如果用户玩该款游戏的时间总是在凌晨或者深夜，那么很可能该用户会有失眠症；不过，也有可能他们正在上夜班。

- **变量与问题本身有关，并且已经通过程序本身追踪和存储。**
  然而需要注意的是，有些可以追踪和存储的变量信息，你很难确定它是否对后期的建模有任何用处。这就是为什么在“特征生成”和“提取”的过程中要尽量多的生成变量。变量的选择工作需要交给后期的“特征选择”工程来做。

- **变量与问题有关，程序可以追踪和存储该变量，但实际上却没有做到。**
  比方说，一个可能重要的变量是“用户是否上传了头像”。然而，你或者你的团队却认为这个变量并不重要，或者你们根本没有讨论过这个变量。这都有可能，任何人或者团队都不会做得面面俱到。这样的变量对建模有很大的帮助，也可以被轻松地记录和保存下来，但最后却没有。然而，对产品事先做好“可用性研究”可以有效地避免忽视掉重要的解释变量（本章的后半部分中，David Huffaker 会介绍有关“可用性研究”的内容）。因为在“可用性研究中”，我们会更多地从用户角度考虑产品设计的方方面面，可以帮助确认需要追踪用户的哪些产品使用行为。

- **变量可能毫无用处，但还是被记录了下来。**
  这太常见不过了，很多记录下来的特征变量都被证明是毫无用处的。但这无伤大雅，因为后期的“特征选择”会过滤掉这些变量，而保留下那些真正重要的变量。

- **变量可能毫无用处，也无从追踪和保存。**
  对于这样的情况就不需要无病呻吟了，既然没有用处，则根本不要追踪和保存，因为你根本不需要它。

现在该回到刚才逻辑回归模型的问题了。用 $c_i = 1$ 表示用户在下一个月的某个时间选择了续订“追龙”游戏。到底是下一个月，下半个月还是下一周，你的团队应该讨论决定，但在建模初始，你和团队不需要太在意这些。等模型的效果稳定之后，可以再商讨这些细节。

通过前章的学习，我们知道逻辑回归的模型形式为：

$$\operatorname{logit}(P(c_i = 1 \mid x_i)) = \alpha + \beta^{\tau} \cdot x_i$$

但你真的会把刚才生成的成千上万的特征变量一股脑全扔给模型吗？即便你这样做理论上没有问题，模型还是会照样运行，但其预测效果想必不会太好，而且运行起来十分缓慢。“特征选择”就是用来从变量集中精选出最为重要的一些变量，既可以提高模型的预测效果，也可以提升模型的运行速度。

关于“特征选择”，Will 强烈推荐大家读一读 Isabelle Guyon 于 2003 年撰写的一篇论文“An Introduction to Variable and Feature Selection”（“变量与特征选择导论”，参见 http://goo.gl/3dz8Ar）。该文从提高模型预测能力的角度详细讨论了如何更好地选择特征变量。变量选择与变量生成以及变量排序有着本质的不同，Isabelle 在文中将主流的变量选择方法分为了三类，过滤型，打包型和内嵌型。下文将分别介绍这三种方法。

7.4.2　过滤型

过滤型是指根据特征变量与因变量之间的某个统计量的值将特征变量从大到小排列出来，再根据排序的顺序过滤掉一批变量。比如，可以用简单相关系数作为统计量。该方法的特点是每个变量都是独立过滤的，不考虑变量间的相互作用。因为其简单有效，通常会用在变量选择的第一步。

过滤型方法的好处是其容易计算，实施起来比较简单。但是，由于其并未考虑变量间的相关信息，可能会有一些副作用。Isabelle 在文中解释说，被过滤掉的两个“不重要”变量单独来看可能确实都不重要，但是放在一起可能会非常重要。变量之间的相互作用往往会将某些不重要的变量联系并组合成一个比较重要的特征。

当然，统计量的类型并不限于简单相关系数。对于线性回归来说，一种较为常见的过滤器可以基于模型参数估计的 p 值或者模型拟合的 R 方。其操作方法是，用每一个特征变量单独建立回归模型并计算该变量参数估计的 p 值或者该模型的 R 方。最后，根据 p 值大小[7]或者 R 方的大小[8]对变量排序（在接下来的“什么选择标准合适”中还有详细介绍）。

7.4.3　包装型

包装型方法是从所有的变量集中（假设变量总数为 n）找到一个最优子集并打包给模型使用。通常子集的大小是一个固定值 k。学过排列组合的同学应该知道，从 n 个变量中选取 k 个变量的可能方法有 $\binom{n}{k}$ 种，因此 k 的选择至关重要。

在应用包装型的变量选择时，有两方面细节需要考虑：其一是选用什么样的算法选择最优子集，其二是选用什么样的“选择标准”衡量一个变量或者变量子集的优劣。

注 7：越小说明该变量越重要。

注 8：越大说明模型的解释能力越强，该变量越重要。

什么算法合适

我们先讨论一种最为常见的变量选择算法，叫作*分步回归*。对于回归模型，该方法的特点是，变量以一种系统性的方式被放入模型（或者从模型中移除）中，同时用类似 *R* 方这样的选择标准记录某个变量被放入模型（或者从模型中移除）时对整个模型效果的影响。通常的分步方式有三种：向前搜索法、向后消除法和双向搜索法。

- *向前搜索法*

  向前搜索是指先建立一个没有任何变量的回归模型[9]，每一步都从变量集合中选择一个变量加入到模型中，选择的标准是看哪个变量可以最大程度地提高模型的拟合或者预测效果。变量被加入到模型之后就不会在从模型中移除，下一步总是从剩下的变量集合中找出一个最佳变量加入到模型当中。每次只添加一个变量，直到所有剩下的变量都不能提升模型时候，模型的向前搜索即停止。

- *向后消除法*

  从搜索的方向上来说，向后相除法与向前搜索法正好相反。刚开始需建立一个包含所有变量的回归模型（称作全模型），然后每次从变量集合中移除一个变量，移除该变量的效果基于其可以最大程度上提高模型的拟合或者预测效果。变量被移除后就不会再返回模型中去，直到移除任何一个变量都不会提高模型的拟合或者预测效果时，模型的向后消除即停止。

- *双向搜索法*

  由于向前和向后消除法的特点是每次只加入或者移除一个变量，而一旦变量被加入就不会再被移除，或者一旦被移除就再也不会返回模型当中。这样的算法对于可能过于绝对，双向搜索法就可以用来解决这个问题。

什么选择标准合适

选择变量的标准有很多，作为一个数据科学家，你还需要在众多选择标准中做出选择。

虽然每一个选择标准都有相应的理论可供参考，但更多的时候，选择起来还是相当主观的。一种可行的选择方法叫作“试错法”：尝试几个不同的选择标准，看模型在哪个标准下表现的更加稳健。不同的选择标准会选出截然不同的模型变量，因此你应该对主流的选择标准有所了解。

- *R方*

  R 方的定义公式为：

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

注 9：意思是，只有一个截距项的线性模型。

线性模型的 R 方大小代表了数据中有多少的信息可以被该模型拟合。

- p值

  在线性模型的参数估计中，通常用 p 值评估一个参数的估计值是否显著。对于某个参数 β，通常设定一个原假设：$\beta=0$，β 估计值的 p 值就代表了在原假设成立的条件下，出现观察样本（通常以估计值的 t 值表示）或者更极端样本的概率。因此，如果 p 值越小则代表观察样本支持原假设的力度越小。也就是说，p 值越小（比如说小于 0.05）我们就可以以相当大的概率（95%）认定原假设并不成立，β 显著不为 0。

- AIC（赤池信息量准则）

  赤池信息量准则的定义公式为：

$$\mathrm{AIC}=2k-2\ln(L)$$

  其中 k 是模型中的参数个数，$\ln(L)$ 是最大似然函数值。AIC 越小模型的效果越好。

- BIC（贝叶斯信息量准则）

  贝叶斯信息量准则的定义公式为：

$$\mathrm{BIC}=k\cdot\ln(n)-2\ln(L)$$

  其中 k 为模型中的参数个数，n 是样本数量（在“追龙”的例子中，就是用户数量），$\ln(L)$ 是最大似然函数值。与 AIC 一样，BIC 越小代表模型的效果越好。

- 熵

  关于熵，我们将在 7.4.4 节中详细介绍。

实际操作

因为分步回归本身的特点，选出的最佳变量子集很容易过拟合数据：模型的样本内拟合效果很好，但是样本外的预测效果却不然。

在应用变量选择标准时，并不需要在每一步都重新训练一遍模型，这会相当费事。由于每个选择标准都可以写成函数的形式，根据函数的泰勒展开式，可以从理论上表示出选择标准如何随着变量个数的改变而改变。

最后，我们还是想提一下，领域专家应该在变量选择过程中起一定的作用，在用算法选择变量之前，不妨咨询一下他们的看法。

7.4.4 决策树与嵌入型变量选择

决策树方法的吸引力在于它非常直观。除数据科学领域以外，人们在日常生活中做决定时，也可以将大问题分解成一系列小问题，逐个解决。比如下图 7-3，一位大学生在决定自己的时间分配问题时所用到的决策树。

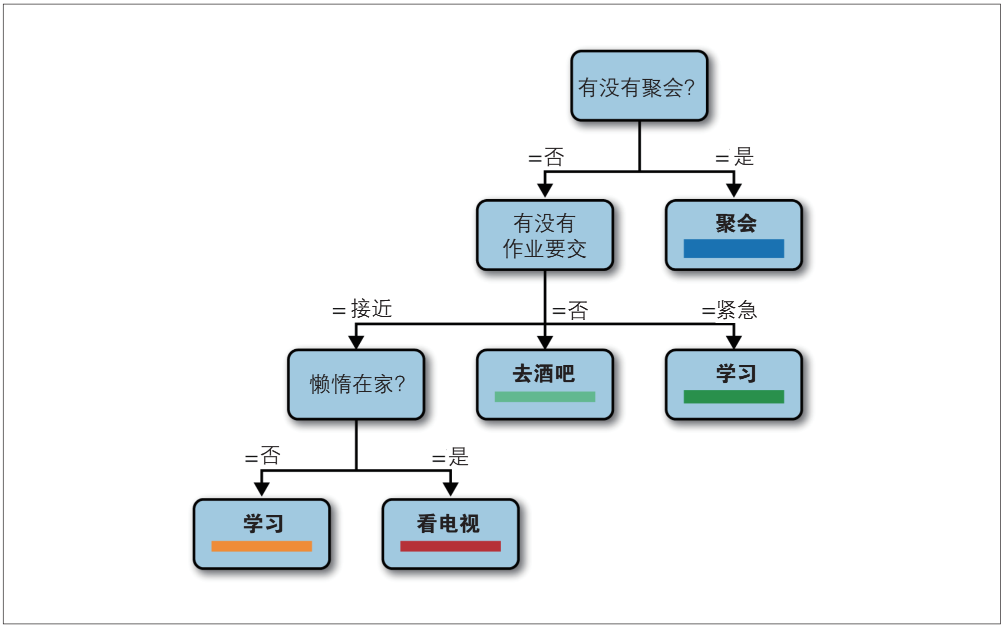


图 7-3：一个大学生在解决自己的时间分配问题时用到的决策树（原图摘自 Stephen Marsland 的著作 *Machine Learning: An Algorithmic Perspective*（《基于算法的机器学习》，Chapman and Hall/CRC），并获得了作者的许可（另见彩插图 7-3）

该决策树的整体结构取决于好几个元素：今天有没有派对、今天有没有作业要交、今天的心情如何，等等。可以看出，决策树的结构清晰、导向明确、非常容易理解，因此具有十分优良的可解释性。这也是决策树如此受欢迎的主要原因。

在数据科学中，决策树一直用来处理分类问题。以之前“追龙”的用户分析问题为例，我们想预测用户下一个月是否会续订。这是典型的分类问题，预测变量只取两个值：“1”代表用户会续订，“0”代表用户不会续订。要想预测用户下一个月续订行为，需要考虑很多因素（比如用户杀死龙的数量、用户的年龄性别、用户在游戏上花去的时间等）。决策树告诉我们可以把这些因素对预测变量的效用用如图 7-3 所示的结构图表示出来，最终得到类似图 7-4 的决策树。

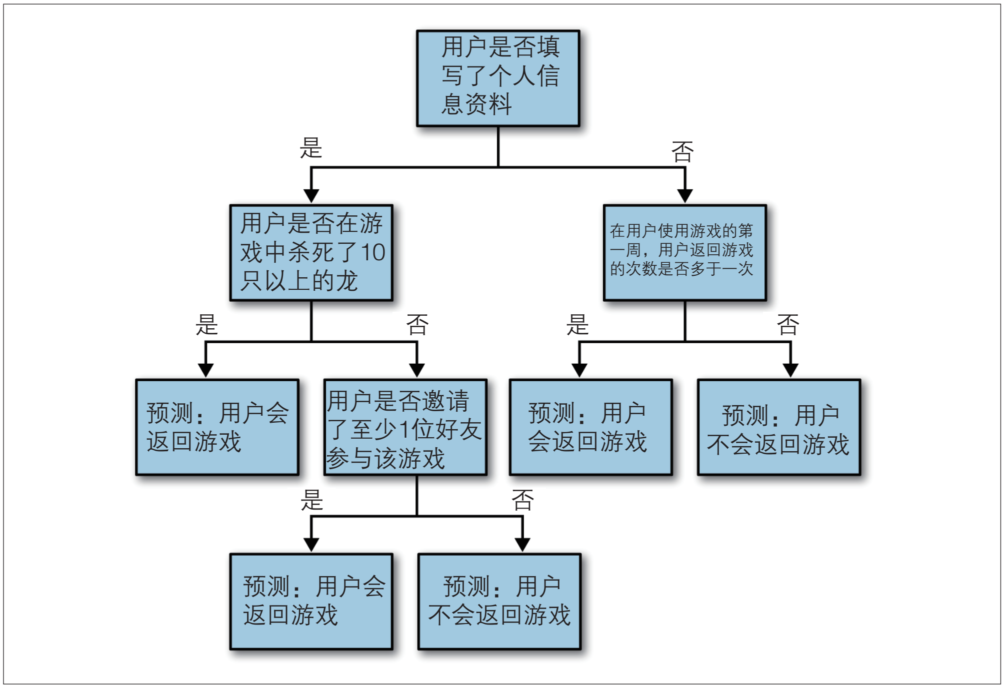


图 7-4："追龙"问题的决策树图

那么一个自然的问题就是：如何把这些变量合理地放在决策树上呢？哪些先放、哪些后放、什么时候停止摆放，等等。的确，建立决策树有诸多细节需要考虑。变量的摆放和拆分要基于数据中的信息，而不是拍脑袋瞎猜。总体原则是，在每一步，都先放上"信息量最大"的变量，从上往下地搭建决策树。因此，一个关键性的问题便是，如何认定一个变量的"信息量"大小呢？

为了说明这个问题，我们用 X 表示数据集中的特征变量，并假设其只能拆分为两类：0 和 1。$p(X=1)$ 以及 $p(X=0)$ 分别代表两类的概率值。

7.4.5 熵

在信息论中，熵用来定量表示一个特征变量所包含的信息量，其定义如下：

$$H(X) = -p(X=1)\log_2(p(X=1)) - p(X=0)\log_2(p(X=0))$$

$H(X)$ 代表变量 X 的熵值。注意，当 $p(X=1)$ 或者 $p(X=0)$ 时，X 的熵为 0。这同下面的极限性质一致：

$$\lim_{t\to 0} t \cdot \log(t) = 0$$

因此，只要 X 取两类中任何一类的概率为 0，则 X 的熵值为 0。此外，由于 $p(X = 1) = 1 - p(X = 0)$，因此 X 的熵关于 0.5 对称，并在 $X = 0.5$ 处取得最大值 1。图 7-5 是 X 的熵与 $p(X = 1)$ 的函数图，$H(X)$ 的对称性和极值点一目了然。

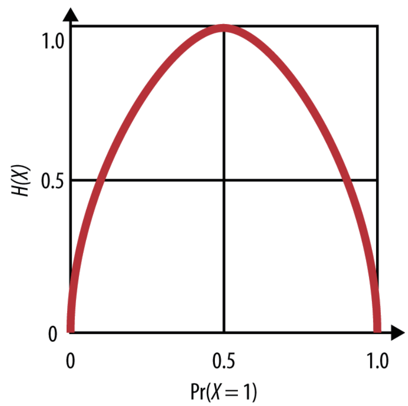


图 7-5：变量 X 的熵值图

从数学上定义的熵值很难被直观地理解，就连“熵”这个词本身就很容易让人抓狂。它到底指的是什么？我们说过，熵代表一个变量所包含的信息量的大小，那上面关于熵的定义公式与信息量大小到底有何关系？

信息量大小与变量的不确定性有着直接关系，一个变量的不确定性越高，其熵值越大[10]。这里不妨举个简单的例子。假设 X 表示一个出生婴儿的性别，$X = 1$ 表示男孩，$X = 0$ 表示女孩。从概率论的角度来说，生男生女的概率是等同的，因此 $p(X = 1) = p(X = 0) = 0.5$。这对于我们来说，基本上没有任何信息含量，也就是 X 具有高度的不确定性，因此其具有大的熵值。这与图 7-5 的结论不谋而合，当 $p(X = 1) = 0.5$ 时，X 的熵达到最大值 1。

相反，如果用 X 表示沙漠中某天下雨的概率，从常识上来讲，既然是沙漠，那么下雨的几率肯定非常小，甚至接近于 0。对于这样的变量 X，由于其不确定性很低，因此熵值也很低。从图 7-5 来看，其熵值接近于 0。

既然熵可以代表一个变量所包含的信息量，我们便可以以它为目标函数，最优化模型的参数。比方说，用 X 表示用户是否续订我们的产品，那么我们必然想知道什么样的变量对 X 的影响最大。这里我们需要定义*信息增益*（Information Gain, 用 IG（X, a）表示）的概念，对于变量 X，在给定一个变量值 a 的情况下，X 熵值的减少量：

$$IG(X,a) = H(x) - H(X \mid a)$$

注 10：在信息论中，这称作系统的混乱程度。

$H(X \mid a)$ 代表 X 给定属性 a 的条件熵。假设 a_0 是属性 a 的实际属性值，那么 $H(X \mid a = a_0)$ 表示在该属性值给定条件下 X 的条件熵：

$$H(X \mid a = a_0)$$
$$= -p(X = 1 \mid a = a_0)\log_2(p(X = 1 \mid a = a_0)) - p(X = 0 \mid a = a_0)$$
$$\log_2(p(X = 0 \mid a = a_0))$$

假设属性 a 有 n 个属性值，那么 $H(X|a)$ 的计算公式为：

$$H(X \mid a) = \sum_{a_i} p(a = a_i) \cdot H(X \mid a = a_i), \quad i = 1, 2, \cdots, n$$

用通俗的话讲，X 给定属性 a 的条件熵指的是在知道属性 a 的取值之后，X 熵值的减少量。因为熵代表了不确定性，也就是说，在知道 a 之后，我们对变量 X 有了新的认识，对它的不确定性减少了。从属性 a 的角度来说，就是它为我们更进一步了解 X 所带来的额外信息量。

有了熵和信息增益的概念之后，我们就可以顺利地搭建决策树了：在摆放变量时，总是优先摆放信息增益量最大的变量，因为它能够为我们了解 X 带来最大的信息量。

7.4.6 决策树算法

决策树是一种迭代算法，先从第一个根节点开始选择变量进行拆分，直到所有的变量都已用尽，或者在某节点上只能对等拆分[11]时，停止迭代。在每一个节点处，都选择可以最大化信息增益的变量用于拆分。

一个完整迭代的决策树往往会过拟合，为了解决这个问题，可以采用“事后剪枝法”。也就是说，为了防止决策树过于复杂，在构建好完整的树之后，在树的某个节点处将树剪短。剪短的决策树往往具有更好的外推扩展能力。

决策树中隐藏了一个变量选择的过程，我们称为“嵌入型变量选择”。这也意味着，决策树模型不需要使用传统的“过滤型”或者“包装型”的方法选择变量。因为，当模型使用信息增量选择变量进行拆分时，已经自动地选择了对模型本身而言最为重要的变量。因此，变量的选择过程已经巧妙地嵌入了决策树模型的搭建过程。

回到刚才“追龙”的例子当中，我们已经搜集了大量的用户数据，并且生成了许多特征变量。我们关心的目标变量是用户在下一个月是否会选择续订该游戏。`rpart` 是 R 中做决策树模型的标准软件包之一。利用其中的 `rpart` 函数，我们可以迅速搭建一个决策树模型，并画出相应的决策树图。里面的代码可供参考：

```
# 利用 rpart 函数建立分类树模型
library(rpart)
```

注 11：拆分后的两类包含的数据量相等。

```
# 运行分类树模型
model1 <- rpart(Return ~ profile + num_dragons +
num_friends_invited + gender + age +
num_days, method="class", data=chasingdragons)

printcp(model1) # 展示模型结果
plotcp(model1) # 交叉验证结果可视化
summary(model1) # 转换为二元变量时阈值选择的详细结果

# 分类树可视化图
plot(model1, uniform=TRUE,
      main="Classification Tree for Chasing Dragons")
text(model1, use.n=TRUE, all=TRUE, cex=.8)
```

7.4.7 如何在决策树模型中处理连续性变量

[12] 软件包在搭建决策树模型的时候已经为我们考虑到了连续性变量的情形，并自动进行了最优离散化的操作。如果没有软件的帮助，你需要弄明白如何最优地离散化连续性变量，并为决策树所用。

离散化关键在于阈值的选取。比如变量 X 指的是用户在游戏中杀死龙的个数，若选取 10 作为阈值，则 X 可以分为一个包含两类的二元变量（1 代表杀死龙的个数大于 10，0 代表小于 10）。被离散化之后的变量可以按照刚才讨论的方式摆放到决策树中，这不是问题。阈值的选取当然不能随意化，这会对模型的效果有着直接影响。

因为有众多的可能性，比方说，离散化为二元类还是多元类，以多大的间隔挑选阈值等。阈值的选取是一个相当棘手的问题。对它的选取，本身可以视作决策树模型的一个子模型。至于到底如何选取，这样取决于数据和问题本身的特点。

泰坦尼克号乘客的生存模型

关于决策树模型，Will 推荐我们看一看 BigML 网站上有关泰坦尼克号乘客生存问题的决策树模型（见图 7-6，参考 http://goo.gl/YsyWJW）。原始数据和源代码都来自于 Encyclopedia Titanica（http://www.encyclopedia-titanica.org/）。图 7-6 只是该决策树模型的一小部分，Encyclopedia Titanica 提供了更为详细的交互化决策树图，感兴趣的读者不妨去他们的网站看一看。

注 12：译注：决策树中的每个节点都会被拆分，因此对于离散型变量来说，拆分操作可以进行得非常自然：只需要按照变量的类别拆分就可以。如果需要在决策树中使用连续性变量来说，其关键就在于如何将变量离散化，选择若干阈值将变量拆分到一些子区间中，并以区间为类。

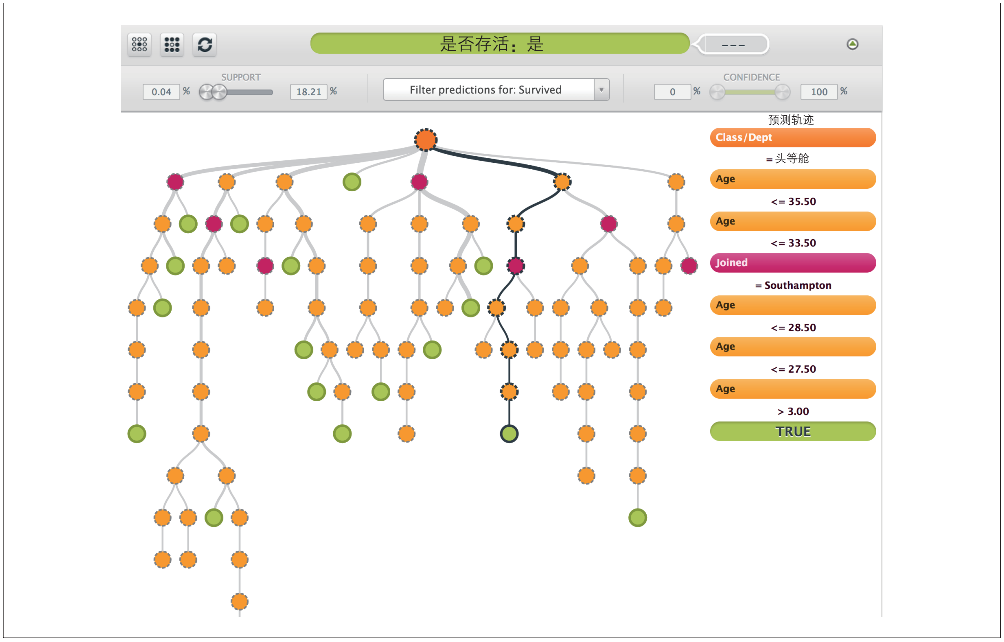


图 7-6：泰坦尼克号乘客生存模型的决策树图（另见彩插图 7-6）

7.4.8 随机森林

随机森林是决策树模型的拓展，它基于 bagging（袋装）模型。bagging 的全称是 bootstrap aggregating（解靴集成法）。该方法可以显著地提高模型的精度和稳健性，但会牺牲决策树模型本身的最大优点：可解释性。人们经常批评随机森林的黑匣子特征，认为从模型形式上来说，其基本不可以解释。然而，从模型设置来看，随机森林模型非常简单，它只有两个参数：N 代表森林中决策树的个数；F 代表每棵树上使用的（随机选取的）特征变量个数。

在深入了解随机森林模型之前，我们有必要先了解一下 bootstrapping（解靴法）。解靴是一种可放回的抽样方式，一个*解靴样本*（bootstrap sample）就是从原数据中有放回取处的 n 个数据，因为抽样是可放回的，因此一个数据可能会被抽取多次，一些数据很可能一次也不会被抽到。至于 n 的大小，一般取所有数据量的 80%，当然这没有严格的规定。对于随机森林来说，n 可以理解为第三个参数。

建立随机森林模型，一般遵循以下两个步骤。

(1) 从原始数据中抽取 N 个不同的解靴样本，每个样本都建立一个决策树。每个决策树只随机使用 F 个特征变量。

(2) 每个决策树的搭建都遵循之前讨论的原则，包括变量的选择，连续性变量的处理等。

每一个决策树都可以进行“事后剪枝”以避免过拟合，但对于随机森林模型来说，这完全没有必要。随机森林的一大特性就是对过拟合的免疫性：模型本身就吸收了数据中的特质方差，因此即使不做单个决策树的“事后剪枝”，随机森林模型也不会过拟合。

下面提供一段随机森林模型的 R 代码：

```
# 作者：Jared Lander
#
# 这里使用的数据是来自 ggplot2 中的 diamonds 数据集
require(ggplot2)

# 载入和大致观察一下 diamonds 数据
data(diamonds)
head(diamonds)

# 画出直方图，在直方图横轴 $12 000 的位置加画一条竖直线
ggplot(diamonds) + geom_histogram(aes(x=price)) +
geom_vline(xintercept=12000)

# 构造一个包含 TRUE/FALSE 的二元变量
# 用来表示该变量的取值是否超过一个阈值
diamonds$Expensive <- ifelse(diamonds$price >= 12000, 1, 0)
head(diamonds)

# 删除 price 列
diamonds$price <- NULL

## 此处需要加载 glmnet 包
require(glmnet)
# 创建预测变量矩阵
# 其中最后一列变量被剔除，因为它是模型的因变量
x <- model.matrix(~., diamonds[, -ncol(diamonds)])
# 构建响应向量
y <- as.matrix(diamonds$Expensive)
# 运行 glmnet
system.time(modGlmnet <- glmnet(x=x, y=y, family="binomial"))
# 画出模型的参数图
plot(modGlmnet, label=TRUE)

# 这里设定一个随机数种子值
# 目的是为了结果的可重复性
# 感兴趣的读者可以多尝试运行几次该程序，得到的结果应该是一样的
set.seed(48872)
sample(1:10)

## 决策树模型
require(rpart)
# 构建一个简单的决策树
modTree <- rpart(Expensive ~ ., data=diamonds)
# 决策树模型分叉可视化
plot(modTree)
text(modTree)
```

```
## bagging 模型（全称是 bootstrap aggregating）
require(boot)
mean(diamonds$carat)
sd(diamonds$carat)
# 下面的函数对均值进行 bootstrapping 操作
boot.mean <- function(x, i)
{
    mean(x[i])
}
# 这样我们便可以计算均值估计的方差了
boot(data=diamonds$carat, statistic=boot.mean, R=120)
require(adabag)
modBag<- bagging(formula=Species ~ ., iris, mfinal=10)

## boosting 模型
require(mboost)
system.time(modglmBoost <- glmboost(as.factor(Expensive) ~ .,
            data=diamonds, family=Binomial(link="logit")))
summary(modglmBoost)
?blackboost

## 随机森林模型
require(randomForest)
system.time(modForest <- randomForest(Species ~ ., data=iris,
            importance=TRUE, proximity=TRUE))
```

特征选择有必要吗？

对于特征选择，有些人觉得没有必要，他们常说：与其费力气做特征选择，还不如花点时间多捞点数据。这也不无道理，比如在用过滤型的方法做变量选择时，如果以简单相关系数为标准，很可能会选出一些与目标因变量高度相关的变量，但他们本身可能毫无关系。在预测标准普尔指数的收益率时，人们发现孟加拉国的黄油产量与其有很强的相关性，但这明显是伪相关。类似的例子还有很多，这些都是由于相关系数本身的限制所造成的。然而，如果有更多的数据，类似这样的伪相关关系可能就没那么重要了。

但是从另外一个角度来看，变量选择其实至关重要。统计模型都会有“偏差 - 方差折中效应”，也就是说越简单的模型可能方差很小，但是其模型偏差很大，而越复杂的模型可能越精确，也就是其偏差可能很小，但是模型方差很大，扩展能力很弱，容易引起模型过拟合。一个好的模型其实就是在偏差与方差中寻求一个最佳平衡点。因此，过多的数据，过多的变量并不会带给我们更好的模型。变量选择对于建模来说，是十分必要的。

7.4.9 用户黏性：模型的预测能力与可解释性

众所周知，决策树模型有着优良的可解释性。也就是说，模型建立之后（要保证其预测能力尚可），可以通过决策树图的形式解析模型。从模型中，可以发现自变量与因变量之间

的逻辑联系。

但是，从决策树图中我们可能会得到这样的逻辑联系：用户在第一个月玩该游戏的时间越长，其在下一个月续订的可能性越大。如果费尽心思建立的模型只能得到这样常识化的解释，分析人员肯定会非常失望。即使不用模型，我们也知道一个人玩游戏的时间越长，代表他越喜欢这款游戏，其续订的可能性当然更大。但是如果模型能够告诉你，在游戏开始的 5 分钟插播广告会减少客户续订的概率，但是如果在用户玩了一个小时游戏之后再插播广告则不会显著地影响客户续订的概率。这时候模型才会显得比较有魅力。因为你从其中得到了你原本不知道的信息，这条信息明确地指示你，在游戏刚开始的 5 分钟切莫插播广告。这样的模型解释性才是真正有用的。当然，为了证实用户是否确实不喜欢在游戏刚开始的 5 分钟内看到插播的广告，还需要进行 A/B 测试（见第 11 章）。但是最起码，模型指出了可能改善用户体验的方向，这才是我们建模的目的所在。

模型的解释可能充满陷阱。这里我们有必要把特征变量分成两类，一类是由用户的行为产生的行为变量（比如用户一个月玩了 10 次该游戏），另一类是游戏开发人员的行为变量（比如每天插播广告 10 次，或者龙的颜色从绿色改成了红色等）。在解释模型的时候，要注意这两种类型变量之间可能存在的相关 / 因果关系及其对模型解释力的影响。比如说，模型的结果显示用户在游戏中得分的高低与其续订游戏的可能性大小有很强的相关关系。那么为了招揽客户，吸引他下个月继续续订，我们是否可以通过给用户增加游戏得分的方式获取更多的用户续订呢？显然不能！用户在游戏中得到的分数与他的续订行为只存在相关关系，而并非因果关系。其中有一个潜在的干扰变量悄然地联系着用户的得分高低与他的续订行为。这个干扰变量就是用户对游戏的喜爱程度，或者上瘾程度。因为用户喜欢这个游戏，玩上瘾了，他才会取得高分，他才会愿意在下个月开始的时候继续续订这个游戏。因此，即便通过变量选择我们可以更好的解释模型，但是真正与改善用户体验相关的，是在考虑到用户的行为变量的影响之后，如何改变开发人员的行为（比如，插播更少的广告，而不是提高用户的游戏得分）。

7.5　David Huffaker：谷歌社会学研究的新方法

David 的工作重点是有效地把定量与定性研究、大数据与小数据研究结合起来，充分发挥各自的优点。大数据研究应该以小数据为起点，先在小数据上建立对问题的基本认知和把握，再延伸到对大数据的整体研究上。反之亦然，在大数据上发现的数据特征，也应该及时地回溯到小数据上进一步验证审查，在一小部分样本人群中做“可用性研究”。此外，也可以用小数据里发现的新东西为大数据特征润色。在小数据上进行的探索性分析也应该联系到大数据上，并与现有的学术文献成果结合，在大数据中找到对应点。这种定量与定性、大数据与小数据的遥相呼应，应该成为数据科学研究的典范模式。

Rachelz 在谷歌时曾与 David 共事，他们当时的合作非常成功。由于他们知识构成相互补

足，当他们在 Goolge+ 项目（谷歌的社交应用）与一些优秀的软件工程师和计算机科学家一起合作时，总是会碰撞出创意的火花，并取得了很大的成功。David 作为一个社会科学家，为团队带来了社会学分析的视角。对于在线社交行为的定量分析与理解，他也同样非常在行。他博士毕业于美国西北大学，研究方向有关媒体、科技与社会。在我们的课堂上，David 与同学们分享了谷歌的社交研究团队的工作方式，他着重提出，一个优秀数据科学团队应该把定量与定性研究、大数据与小数据研究相结合才能取得成功。

谷歌的工作方式非常开放，不同背景的人组成团队，攻克同一个项目，这种混搭的方式会产生巨大的生产力。学术研究与项目开发的界限在谷歌已经被极大程度地弱化，很多项目都带有强烈的学术研究色彩。谷歌团队在 2012 年 6 月就他们的研究和工作方式甚至发表了一篇名为“Google’s Hybrid Approach to Research”（“混搭研究在谷歌”，参见 http://goo.gl/ejtPw2）的论文。在谷歌的产品开发队伍中，研究团队是重要的组成部分。谷歌的产品总是在不断实验、不断完善的。工程师们从产品立项的第一天起就尽量保证代码的质量，产品在小范围内实验后，会慢慢地开放给大众用户。因为谷歌的产品总是拥趸众多，因此他们不可能总是从头再来，产品开发的每一步都要脚踏实地。在小范围客户中验证产品可行性之后，再慢慢地扩大用户群，并在此过程中不断获取用户反馈并及时改善产品，这就是谷歌的产品开发哲学。

7.5.1　从描述性统计到预测模型

David 指出，作为数据科学家，我们的工作决不能只停留在描述性统计阶段，应该勇于跨入深水区，设计实验，研究变量之间深层次的关系。这需要我们敢于把工作重心从描述性统计转移到预测模型上来。

他举了一个在谷歌内部发生的例子。Google+ 的一个重要功能叫作“圈子”，在设计之初，谷歌的研究人员已经知道人们在分享照片或者新闻时有着强烈地选择性。比如，人们更倾向于把照片分享给亲人，一些八卦只会与自己的好朋友分享。这就是谷歌设计“圈子”的初衷。然而，即便这个功能听起来似乎能够满足人们的需求，但是谁也不能肯定用户一定会喜欢上这个功能。人们分享的动机和方式因人而异，用户的心理其实很难揣摩。

为了验证产品的可行性，谷歌采用了一个*混合研究方式*，使用了形形色色的研究方法，包括小型的定性研究，还有大规模的定量研究，最后确认了产品的可行性和应该具备的主打功能。

谷歌其实主要采取了问卷调查和访谈的研究模式，但细节上更为复杂。为了确定什么样的“圈”名最流行，谷歌随机挑选了 10 万名用户，从其中挑选出 168 位最为活跃的用户，请他们填写一份调查问卷，并对其中的 12 位用户进行了细致访谈。为了消除取样时的选择偏差，他们对访谈的深度相应地也做了调整。

他们发现，大多数人都会选择性地分享，大多数人都会使用“圈子”的功能，而“圈子”

的命名方式大多跟工作或者学校的名字有关，有时候从“圈子”的名字就可以看出这帮人是群死党还是群刚认识的朋友（其中一个圈子的名字叫“一辈子老顽童组”，很明显是一帮死党老朋友，另一个圈子的名字叫作“新东方同学”，很明显是一帮刚从英语课上认识的学生）。

在问卷中，他们询问了用户为什么会分享内容。回答基本可以分为三类：其一是因为想展现自我，包括自己的经历、观点等；其二是用户具有参与某个对话的欲望；其三是一些天生就乐于分享的人。

他们还询问了用户在选择分享对象时，都有哪些考虑。回答也可以分为三类：其一是私密性：有些信息只会分享给最亲的好友；其二是相关性：一般信息只会分享给与信息相关的人，或者对信息感兴趣的人；最后是覆盖度：有些人分享只为了尽可能大的覆盖所有的好友，甚至公众，以扩大自己的影响。

总而言之，用户的分享行为确实具有相当的选择性，主要的影响因素是分享的内容和分享的对象。针对这个特点，谷歌在打造“圈子”的产品主打功能时，应该着重于优化用户控制分享的内容以及对象上的体验。只要做好了这些用户最关心的功能，最终的产品才会取得成功。

思维实验：大规模社交网络分析

在第 10 章中，我们会和 John Kelly 一起详细学习社交网络分析的内容，现在让我们稍微热身一下。我们刚刚已经看到了 Google+ 团队在设计产品时，针对用户的选择性分享行为开展了卓有成效的用户可行性研究。想一想，如何把该研究的结论应用到大数据层次上？现在较为流行的做法是使用网络图，对于 Google+ 来说，每一个用户代表一个节点，“圈子”内的用户之间可以用有向连接线相连。在小范围的调查结束之后，你需要想一下如何将调查的结论拓展到大数据上：需要记录哪些数据、构造什么假设并设法在网络图中找到答案。

作为一个数据科学家，应该拓展思维空间，用不同的结构和形态表示数据，这样你会发现，即使是网络图，也可以画出各种不同的模样，代表数据中不同的特征。

下面是网络的几种不同形态。

- 每一个节点都代表网络世界中（Second Life，参见 http://secondlife.com/）的用户，节点之间的连接线代表用户之间的互动。因为用户之间的互动可以有许多不同的形式，因此也应该允许节点之间的连接线有不同的类型。
- 每一个节点代表一个网站，连接线代表网站之间的超链接。
- 节点代表一个定理，而连接线代表定理之间的关系（参见这个例子：http://www.math.columbia.edu/~dejong/plaatje.png）。

7.5.2 谷歌的社交研究

谷歌的所有产品都带有强烈地“社交”元素，甚至连谷歌的搜索引擎也是如此：比如说，你搜索某个条目，谷歌会告诉你，你的某个朋友也对这个条目点了赞，这称作社会化标注。通常人们更关心专家们，而不是亲戚朋友朋友所标注的信息。这很简单，假设你准备买一瓶酒，你肯定更想看到品酒专家对某款酒的标注，而不是你的某位亲友。

但是如果问，在品酒专家和亲友之间你更加信任谁，你肯定会选择亲友。其实也就是说，“点赞”的朋友并不一定是亲密的朋友，人们在社交网络上所表现出来的行为和情绪有着强烈的复杂性。到底用什么来度量社会化标注的重要程度，还是要取决于标注本身的特点。

通常，数据科学家喜欢用诸如“点击率”这样的指标衡量其重要程度。

7.5.3 隐私保护

社交网络上最为棘手的问题就是用户的隐私保护问题。谷歌曾经做过一项调查，问题是关于人们对社交网络隐私保护的看法：比如隐私顾虑是否会削弱他们参与社交网络的热情；通常的隐私顾虑都包括哪些内容，等等。

调查结果显示，用户对隐私的顾虑会直接影响他们参与社交网络的热情。这一点都不奇怪，因为社交网络的急速传播性，用户会顾虑他们分享的每一条信息，关心什么时候分享这条信息比较合适，信息还会被哪些好友分享，等等。在社交网络上，我们对信息毫无控制权，当你面对这么多烦恼的时候，你很容易产生负面情绪，对参与社交网络的兴趣大减。

下面列出调查中用户列出的一些顾虑。

- 身份信息窃取
    - 银行卡信息被窃等会导致潜在的金钱损失
- 数字世界
    - 个人信息
    - 个人搜索历史
    - 垃圾邮件
    - 私人大尺度照片（被老板看见就惨了）
    - 不情之请
    - 垃圾广告
- 真实世界
    - 离线骚扰

- 威胁家人
- 被人跟踪
- 失业风险

7.5.4 思维实验：如何消除用户的顾虑

用户的以上顾虑其实细细看来都比较合理，Rachel 课上的同学对这些顾虑展开了头脑风暴，并给出了以下一些解决方案。

- 在产品说明页详细地写明相关的隐私政策协议。谷歌确实这么做了，但好像基本上没有人愿意花时间读这些协议。
- 可以把推荐的内容更加人性化地推送给用户，比如说："尊敬的用户，基于你对某某条目点了赞，我们猜您可能也对这条信息感兴趣。"但是用户对此仍会心存疑虑。
- 可以告诉用户，所有用户数据的有效期为一年。在一年之后所有数据都会从服务器上永久删除。

其实，只要把数据政策透明地呈现给用户，用户的很多疑虑都会消除。但关键问题是，怎么让这些政策更加透明化呢？下面是学生们讨论的几点建议。

- 将数据的使用状态用动态图形或者动画的形式展示给客户。
- 用户需对自己的隐私有控制权。
- 用户可以快速设置相关的隐私选项。
- 设置中往往会存在很多不人性化的"霸王条款"，可以考虑把这些条款做得更加人性化一点。
- 最为理想化的情形是，所有的默认设置都充分考虑到了用户的隐私顾虑，因此用户不需要作任何改动。

David 在最后留了一句话给大家，作为本章的结束：当你向大数据迈进的路上，不要被定量分析蒙蔽了双眼。简单的、基于逻辑的定性分析对于更加深刻地理解数据同样重要。像定性调查分析这样的传统工具，有时候反而能取得奇效。

第 8 章

构建面向大量用户的推荐引擎

推荐引擎，又名推荐系统，是数据科学的典范应用之一。日常生活中，人们或多或少都同推荐系统打过交道，比如在亚马逊买书时，系统会推荐你可能感兴趣的图书，Netflix 会推荐你可能喜欢的电影。因此推荐引擎是向门外汉介绍数据科学及其应用的一个很好的切入点。然而，很少有人会思考隐藏在推荐系统背后的算法细节，更不会有人意识到当他们在网上买书或者评价电影时，相应的数据都自动生成并提供给了推荐系统，这些数据反过来又会提高推荐系统的准确性，为人们推荐更多更好的内容。

之所以称推荐系统为“典范”的数据科学应用，不仅是因为它是一个典型的数据推动的应用，还在于构建该系统时需要一些数据科学的两项必备技能：线性代数和编程。同时，推荐系统还展示了一些看似直观的解决方案在面对大数据时可能会面临的计算量上的挑战。

本章，Matt Catts 将和我们一起分享他在为 Hunch.com 构建推荐系统时所积累的经验。在构建这种直接面向用户的大型推荐系统时，他是基于何种考量做出某些决定的；在面对各种算法的时候他又是如何做出取舍的，我们都将一一揭晓。

Matt 毕业于麻省理工学院，专业是计算机科学。毕业后供职于 SiteAdvisor 公司，随后与人合伙创建了 Hunch 公司（http://hunch.com/），自己担任 CTO。Hunch 是一个为用户作各种推荐的网站。首先，系统会问用户一系列问题（从 Hunch 网站的数据来看，人们似乎很乐意回答网站提出的问题），根据用户对这些问题的回答，Hunch 会为每一位用户构建一个推荐系统。如果用户反过来问系统一个问题，比如“我该买一个什么样的手机？”“这次我该到哪儿去旅游？”，系统就会给出为该用户量身定制的、合理的建议。利用机器学习技术，随着系统的不断学习，这些推荐的质量会变得越来越好。Matt 的工作，就是负责整个推荐引擎背后的技术开发和研究（R&D）。

起初，他们会把问题设计得尽量有趣以争取更高的回答率。后来发现，这些问题的设计应该以尽可能获取最多的用户信息为宗旨。经过不断地调整，他们发现只需要问用户不超过 20 个问题，推荐系统的预测准确率就能达到 80%。有些问题很常见，而有些问题对于用户来说可能会比较出乎意料，比如会问你的性格是争强好胜还是随遇而安、内向还是外向、理性或感性，有点像 MBTI 的人格测试。

后来，Hunch 决定不直接向用户提问，转而向互联网抓取数据，并以 API 形式向外提供服务。服务可以被第三方用来为既定网站提供个性化的推荐内容。无疑，这是一个更好的商业模式，最终 Hunch 成功出售给了 eBay。

Matt 从小就开始写程序，这是他的强项，除了推荐系统，他并非所有领域的专家，而 Hunch 作为一个数据公司，需要跨领域、多学科知识的人才。

因此，Matt 说过一句很经典的话：”一个人不可能完成所有工作，组建数据团队就像召集十一罗汉一样，个个都得身怀绝技。”[1]

8.1 一个真实的推荐引擎

推荐系统的适用范围十分广泛。电影公司会根据用户的观影历史，推荐给客户他们可能会喜欢的电影；在线购书网站根据客户过往的购书记录为用户推荐他们可能喜欢的新书；旅游公司会根据用户以前旅游过的地方，为他们建议一次新的旅程。这样的例子数不胜数。

构建这样的系统虽然方法很多，但总体感觉起来其实都大同小异。本章会为大家介绍如何实现这样一个系统，我们的实现虽然简单，但是麻雀虽小，五脏俱全。

假设有一个用户的集合 **U**，待推荐的项目定义为集合 **V**，如 Kyle Teague 在第 6 章中指出的那样，可以用二分图（见图 8-1）表示两个集合之间的关系，如果用户评价过某件东西，就用一条线将二者连接起来，这种评价有好有坏，评价可以是正面的，可以是反面的；可以是连续的，可以是分散的，我们可以给线加上权重以表示这些连接在某种意义上的“强度”值。评价系统可能会很被设计非常复杂，但这里我们不讨论过于复杂的系统设计。

推荐系统所需要的训练数据集其实就包含在上面的二分图中：主要说来，就是用户的特征变量以及用户对某些商品的偏好数据。有了这些数据之后，基于推荐系统的分析，就可以为用户推荐新的商品了。最终具体的推荐项目就是推荐系统的模型输出结果。

你可能会获取的数据包括一些用户或商品的元数据，比如用户性别，商品的颜色。当用户登录访问你的网站时，需要注册账号，你就可以得到用户的性别、年龄和个人嗜好等数据，还可以让他们选择自己喜欢的三件商品。

注 1：《十一罗汉》是一部有名的美国电影。

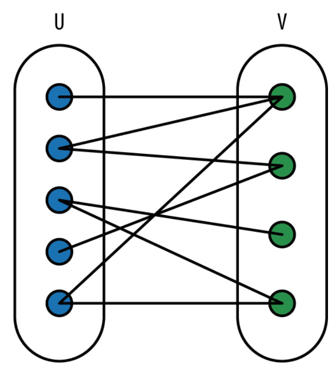


图 8-1：推荐系统的二分图：左侧是用户，右侧是推荐的项目，比如电视节目（另见彩插图 8-1）

用户可以表示为一组特征向量，该向量有时包含用户所有的特征变量（包括用户的元数据以及用户的其他特征变量），有时可能只包含用户的元数据或者只包含用户的喜好数据，这取决于你想用这组特征向量干什么。如果向量只包含用户的喜好数据，那么得到的向量可能会十分稀疏，因为不同用户的喜好会差别很大。在得到每位用户的特征向量之后，可以把他们绑在一起，形成一个大的用户向量矩阵，我们不妨称为 **U**。（之前的用户集合也叫作 **U**，这里我们有点滥用数学符号的嫌疑。）

8.1.1　最近邻算法回顾

第 3 中我们讨论了最近邻算法，这里让我们就推荐系统问题对这个算法稍作回顾：如果你想预测用户 *A* 是否喜欢某商品，你可以问问 *A* 的“邻居” *B* 的意见。如果 *B* 喜欢该商品，那么很可能 *A* 也会喜欢该商品。换句话说，用户 *A* 的相似用户是用户 *B*，用户 *A* 没有看到过或者买过某种商品，而用户 *B* 买过，那么我们就可以向用户 *A* 推荐该商品，因为基于最近邻原理，他的邻居买过此商品，那么他很可能也会对此商品感兴趣。

第 3 章我们说过，最近邻算法严重地依赖于“距离”的定义。对于用户的喜好变量（是一个二元变量）可以用 Jaccard 距离来测度用户之间喜好的相似程度（即“1 – (二人都喜欢的商品个数 / 只有其中一人的商品个数)”）。另外，余弦度和欧式距离也是合理的选择。



到底哪种距离测度最好？

这个问题我们已经提过很多次了，答案是：没有严格、统一的标准。具体问题需要具体对待，我们建议在实际操作中尝试不同的测度，再通过模型评价选定一个较为合理和有效的。

8.1.2 最近邻模型的已知问题

最近邻模型固然是一个不错的模型，它足够简单，模型逻辑也很容易理解：为了给某用户推荐某商品，最近邻模型会先找该用户的“邻居”，并根据这些“邻居”的喜好情况为该用户设计推荐内容。但是有很多已知的问题限制着该模型的适用范围，我们这里列举一些。

- *维度诅咒*

  当数据的维度过高时，即便是最近的“邻居”在高维空间中也会离得很远。

- *过拟合问题*

  如果用 $k = 1$ 的最近邻算法会很容易产生过拟合的问题。因为离你最近的那个“邻居”所能提供的信息十分有限，它本身可能包含大量“噪声”信息。日常操作中，我们很少使用 $k=1$，$k=5$ 是一个较为常见的选择。但是过大的 k 同样也会带来更多“噪声”信息。

- *多重相关问题*

  数据中的特征变量很多，他们中间很多是高度相关的。多重相关问题会带来信息重叠问题，比如年龄大的人往往倾向于比较保守的性格，如果把用户的年龄和保守程度都用在模型中，就会造成某项信息的过度使用。信息的重叠和过度使用都会带来欠佳的模型表现。如果能把数据中的特征变量精简，只采用一些重要的、信息没有重叠的（不多重相关的）的变量，对于提升模型效果是大有裨益的。

- *特征变量之间的相对重要性*

  某些变量可能比另外一些变量更加重要，这是很容易理解的。比如你的年龄对于预测你对商品 1 可能没有其他变量重要，但是在预测你对商品 2 的时候可能又至关重要。根据研究对象的不同，要给不同的变量赋予不同的权重，这对于提高模型效果也通常是有帮助的。至于如何确定权重，可以看看变量之间的协方差矩阵。

- *稀疏性*

  如果特征变量的向量（或者特征变量矩阵）过于稀疏，或者有过多的缺失值，那么也就意味着数据中的绝大多数信息是未知的。如果用 Jaccard 距离这样的测度就很难得到有用的信息。

- *测量误差*

  数据的测量总有误差（也称作报告误差）：因为人们是会说谎的。

- *计算的复杂度*

  计算是有成本的——成本取决于计算问题本身的复杂度。[2]

注 2：近邻模型在数据量较大时的计算成本是很高的。

- 距离测度指标的敏感性

  有些距离指标对于变量的取值范围是很敏感的。比如欧氏距离就有这个问题：年龄变量之间的欧氏距离就要远远大于二元变量之间的欧氏距离。也就是说，由于欧氏距离对变量取值的敏感性，它的应用受到了较大的限制。

- 时变效应

  用户的喜好可能会随着时间的改变而改变，这样模型就不再适用了。比如在 eBay，如果一个用户刚买了一个打印机，那么在短时间之内他可能只想买墨盒。过了一段时间他又会想买别的商品。

- 模型更新成本

  随着新数据的加入，我们需要不断地更新模型。而模型更新的成本往往比较高。

上面所列举的诸多问题中，前两项是最为重要的：维度诅咒和过拟合问题。我们应该如何对付这两个问题呢？我们接下来以线性回归模型为出发点，探讨一下这个问题。

8.1.3 超越近邻模型：基于机器学习的分类模型

我们接下来把机器学习模型简化，针对每一个商品使用单独的线性回归模型用于预测。在某个模型中，我们的目的是给定某位用户的属性变量，预测该用户是否会喜欢该商品。比如，某个模型被单独用来预测你是否喜欢电影 *Mad Men*（《广告狂人》），另外一个模型被单独用来预测你是否喜欢歌手 Bob Dylan（鲍勃· 迪伦）。

用 $f_{i,j}$ 表示用户 i 对商品 j 的喜好情况（或者 j 代表用户 i 的元数据，比如年龄和是否是注册用户等，而 i 则是用户的其他属性变量）。这个表示方法可能会让人有点摸不着头脑，所以你要花时间稍微消化一下，我们在这里把元数据也当作了某种意义上的“商品”。我们之前也有过这样的表述，如果你实在觉得很难理解，也没有关系，继续往下看。当我们说推荐系统可以预测你的喜好时，从模型层面上来说这里的“喜好”是广义的，可能是你的某种属性变量。比如说，也许你在注册时没有填写性别或者我们没有问你的性别是什么，那么“性别”可以作为一个待预测变量，也就是这里的“喜好”。

为了让这种表述变的更容易理解，我们假设每个用户都有三种属性变量，也就是说对于任何用户 i 都有 $f_{i,1}, f_{i,2}$ 和 $f_{i,3}$。为了预测该用户 i 对于某个新商品的喜好（暂时用 p_i 表示该喜好），应用以下的线性回归模型：

$$p_i = \beta_1 f_{1,i} + \beta_2 f_{2,i} + \beta_3 f_{3,j} + \epsilon$$

好消息是，对于这样一个多元线性回归模型的参数估计，我们已经学习过了。参数的估计可以使用线性代数的方法或者最优化的方法。这是线性回归模型的统计推断，我们在之前已经遇到过了很多次。

但是坏消息是，这样的模型是针对单个商品的，也就是说，有多少商品就要有多少个类似的线性回归模型。而且，由于模型都是独立的，因此这样的模型没有考虑进商品之间可能存在的相关关系，这对于数据分析本身来说，就意味着大量的有关商品之间关联的信息被忽略和浪费了。

等等，我们之前说过的变量的权重问题在线性回归模型中得到了很好得解决，因为回归模型的参数就是天然的权重估计值。

但是更严重的坏消息是，这样的模型很容易产生过拟合的问题。从表现形式上来说，如果对某个商品的喜好的数据量不足，那么在该商品的回归模型中，参数的估计值会变得很大。

> 我们刚才说到如果模型中的某些参数的估计值过大，那么就认为中模型出现了过拟合，这里做一下解释。试想在模型中我们使用了两个一模一样的变量，其中一个变量的参数估计值可能为 100 000 而另外一个是 –100 000，从参数估计的绝对值来看，似乎两个变量都十分重要，但我们知道其实其中一个变量是完全多余的。因此在建模之前，我们最好做一些关于参数估计大小上下限的先验假设，一切超过该上下限的参数估计值都是可疑的，很可能意味着模型已经出现了过拟合。

为了解决过拟合问题，可通过设定贝叶斯先验信息的方式强制所有的参数估计值都落在一定范围之内。从模型形式上来看，我们可以惩罚过大的参数估计值。惩罚大参数等同于在数据的协方差矩阵上加了一个先验信息矩阵（参见 http://mathbabe.org/2013/02/24/the-overburdened-prior/）。最后的模型解只取决于一个参数，通常用 λ 表示。

接下来一个自然的问题就是，该怎么选择这个惩罚参数 λ 呢？第 3 章的模型交叉验证的方法也可以用在这里：选取一部分数据作为训练数据，一部分用作模型验证与评价。不停地变换 λ 的值，直到模型的评价标准告诉你已经找到了一个较为合理的值。现实中我们经常使用这样的交叉验证方法，但是却很难保证一定能找到那个最优的 λ 值。

> 有些变量因为取值本身就比较大，如果用惩罚的方法，对它们可能非常不公平。解决这个问题最简单和直接的办法就是在建模之前把所有的变量正则化，我们在第 6 章讲过正则化的概念。这样所有的变量都在同一个水平线上取值，惩罚就变得相对公平了。正则化的操作可以根据变量意义的不同而有所差异，至于到底在正则化中使用什么均值和方差，建模者可以自由选择。从贝叶斯统计的角度来说，正则化操作等同于在模型中添加了某种形式的“先验信息”。

最后一个关于先验信息的问题是：虽然将先验信息添加到模型中，只要 λ 够大，模型一般

都会有唯一的最优解。但是过大的 λ 可能会导致模型中绝大多数甚至全部的参数估计都接近于 0，这样模型本身的实际意义就荡然无存了。

8.1.4 高维度问题

我们已经解决了过拟合的问题，第二个棘手的问题就是数据的高维度问题了：比如刚才的数据中可能包含上百万商品的信息，这样的维度会给模型带来很多问题。解决高维度问题的两大杀手锏是*奇异值分解*（Singular Value Decomposition）和*主成分分析*（Principal Component Analysis），接下来我们会详细讲解。

在讲解其中的理论细节之前，想一想在现实生活中我们是如何通过“隐含变量”（latent feature）的形式达到降维目的的。我们经常说某人很“酷”，而这个概念是很难量化和直接观测到的，我们称为“隐含变量”。而“酷”本身可能是很多因素构成的，比如这个人的行为方式，语音语调等，这些因素往往是可以直接观测到的。因此一个“酷”就涵盖了许许多多可观测变量的信息，这从形式上来说，就达到了降维的效果。

在这个降维过程中发生了两件值得注意的事情：其一是很多变量的被一个隐含变量所涵盖，其二是这个隐含变量不可观测。

从数学上来讲，对于包含很多可观测变量的数据矩阵来说，我们并不知道其中的隐含变量意味着什么、该如何解释等，我们甚至不知道其中有几个“重要”的隐含变量。所有的一切都是由算法和计算机自动完成的。“重要”在这里指的是该隐含变量可以解释数据中绝大部分的方差。如果可以用很少的“隐含变量”就能解释数据中绝大部分的方差，那么这些隐含变量就是“重要”的。

从推荐系统的角度来看，降维的目的是找到用户的“品味”（taste information）。根据每个用户品味的不同，系统可以做出个性化的推荐。因此，“品味”在这里是一个隐含变量，并且提取自关于用户的可观测的变量。

不过，因为很多属性变量都来自于系统在用户注册时的问卷调查数据，也就意味着很多变量是简单的二元变量。而二元变量所包含的信息往往少于连续性变量和多类别变量。Hunch 的研究人员就发现，问卷中多设计一些比对问题（comparison question）能够带来更好的推荐效果。

是时候好好学习一下线性代数了！

如果不太买线性代数的账，或者对于其中诸如“秩”（rank）、“正交”（orthogonal）、“转置”（transpose）、“基”（base）、“张空间”（span）和“矩阵分解”（matrix decomposition）这些概念的几何解释都非常陌生，那么本章之后的内容你一定会感觉一头雾水。

从空间和矩阵的观点看数据，我们能够更深刻地洞悉数据。理解诸如矩阵的变换和子空间等概念对深入理解某些模型的数学细节，或者对于优化某些模型的源代码都大有帮助。矩阵角度的模型洞察力甚至决定了一个数据初创公司的技术实力，这也是 Hunch 能被 eBay 收购的主要原因：他们拥有厚实和前沿的技术实力。如果你觉得自己的线性代数学的不够好，我们推荐大家去听一听 Khan Academy（可汗学院）上的线性代数课。

8.1.5 奇异值分解（SVD）

我们已经热身很久了，下面就让我们直接进入奇异值分解的数学细节。给定一个秩为 k，维度为 $m \times n$ 的矩阵 $\boldsymbol{X}$，根据线性代数的理论，矩阵 $\boldsymbol{X}$ 可以分解成三个矩阵的积：

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^{\tau}$$

其中 $\boldsymbol{U}$、$\boldsymbol{S}$ 和 $\boldsymbol{V}$ 的维度分别为 $m \times k$、$k \times k$ 和 $k \times n$。$\boldsymbol{U}$ 和 $\boldsymbol{V}$ 矩阵内的列是相互正交的，$\boldsymbol{S}$ 是一个对角矩阵。SVD 在线性代数中的标准描述是把 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 称作为方酉矩阵（square unitary matrix），而把 $\boldsymbol{S}$ 称作方阵（rectangular matrix）。我们也沿用这个叫法。之后我们将通过减秩的方法尽可能的近似矩阵 $\boldsymbol{X}$，同时达到降维的目的。关于 SVD 分解存在性的证明可以在奇异值分解的维基百科页面上找到（http://goo.gl/GLS6sG）。

现在让我们把 SVD 分解与之前推荐系统的数据分析问题联系起来。$\boldsymbol{X}$ 对应之前的原始数据集，里面包含用户和商品的信息，其中有 m 位用户和 n 件不同的商品，$\boldsymbol{X}$ 的秩为 k。k 也是 $\boldsymbol{X}$ 中可能包含的隐含变量个数 d 的上限值。d 是 SVD 需要确定的调整参数（tuning parameter）。

SVD 分解中的矩阵 $\boldsymbol{U}$ 的每一行代表用户，矩阵 $\boldsymbol{V}$ 的每一行代表商品。$\boldsymbol{S}$ 矩阵的对角线上的元素叫作“奇异值”（singular value），他们的大小衡量的是相应位置上隐含变量的重要性：最重要的隐含标量对应最大的奇异值。

8.1.6 关于SVD的重要特性

因为矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 各自的列向量之间是相互正交的，因此可以根据奇异值从大到小的顺序将他们变换位置。因为奇异值的大小代表着相应位置上隐含向量的重要性，因此在排序之后可以考虑扔掉奇异值很小的那部分矩阵而只保留奇异值较大的那部分，它们代表了原数据矩阵的大部分信息。

也就是说，我们把 $\boldsymbol{S}$ 矩阵的对角线元素按照从大到小的顺序重新排列了之后扔掉了其右下方的大部分元素，同理 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 也相应地做了重新排序和舍弃。如果 d 是一个远小于 k 的数，那就代表我们舍弃了 $\boldsymbol{U}$、$\boldsymbol{S}$ 和 $\boldsymbol{V}$ 的绝大部分元素，只保留了对应大奇异值的那一小部分元素。计算机图像学中经常讲到的“压缩”（compression）概念就是如此。原矩阵 $\boldsymbol{X}$ 的绝大部分信息被保留了下来，但是矩阵的规模却缩小了很多。矩阵 $\boldsymbol{S}$ 的对角线元素具有明

确的含义，矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 中的值也同样有很好的解释。

这样的矩阵分解对于推荐系统来说到底有何用处呢？设想矩阵 $\boldsymbol{X}$ 中已经填满了用户对商品的打分数据。（通常来说，矩阵中最好不要出现打分为 0 或者没有打分的情况，没有打分就意味着缺失值，而 SVD 不能处理有缺失值的矩阵）应用 SVD 分解完矩阵之后，我们想要的并不是三个独立的分解矩阵，我们的目的其实是为了预测新的用户对某商品的喜好。而这个预测可以用矩阵 $\hat{X}$ 来表示，而它恰好就是我们刚才讨论过的 $\boldsymbol{X}$ 的近似矩阵。

在 8.1.2 节中我们讲到了最近邻模型的很多已知的缺陷，比如缺失值问题和计算量大的问题。SVD 也有这两个方面的问题。SVD 涉及的矩阵分解对计算机的计算能力要求颇高，因此对于大型矩阵 SVD 会显得力不从心。我们应该想一想如何改进 SVD 的计算速度问题。

8.1.7　主成分分析（PCA）

保留 SVD 中的 $\boldsymbol{U}$ 矩阵和 $\boldsymbol{V}$ 矩阵，如果我们可以抛弃 $\boldsymbol{S}$ 矩阵而只通过这两个矩阵近似原矩阵 X，则有：

$$\boldsymbol{X} \equiv \boldsymbol{U} \cdot \boldsymbol{V}^{\tau}$$

如果这个近似是可行的，那么我们总是希望 $\boldsymbol{X}$ 与 $\boldsymbol{U} \cdot \boldsymbol{V}^{\tau}$ 之间的误差尽可能小。这是一个最优化问题，我们希望最小化他们之间的误差平方和：

$$argmin \sum_{i,j} (x_{i,j} - u_i \cdot v_j)^2$$

此处，u_j 表示 $\boldsymbol{U}$ 矩阵对应用户 i 的某行，v_j 对应 $\boldsymbol{V}$ 矩阵中商品 j 的某行。同样，$\boldsymbol{V}$ 矩阵的行上可以包含用户的元数据变量，比如年龄变量。

那么点积 $u_j \cdot v_j$ 就代表用户 i 对商品 j 喜好程度的估计值，我们当然是希望其与真实值越接近越好。

也就是说，你想找到一组最优的 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 矩阵，以最小化预测值与实际观测值之间的平方误差。实际观测值代表你已经观测到的，对其有充分了解的值，那么这里的逻辑就是，如果它们对于已经观测的值有良好的表现，那么对于没有观测到的值，其表现也会十分不错。你应该已经对这个逻辑相当熟悉了——这就是均方误差的概念，我们之前在线性回归里已经仔细地阐述过了。

这里唯一需要确定的参数就是 d，它的数值代表了隐含变量的个数。矩阵 $\boldsymbol{U}$ 的行代表用户，列代表隐含变量；而矩阵 $\boldsymbol{V}$ 的行代表商品，列代表隐含变量。

到底如何选择 d 呢？在我们的例子中，一个合适的 d 大概在 100 左右。我们之前说过在用户注册填写问卷调查的时候，20 个问题就足够了解该用户了，因此 100 似乎看起来有点过大了。d 的大小可以人为设定，只要不出现过重的计算负担，我们建议将 d 设在 100 左右。

得到的隐含变是原矩阵在 n 维空间上的基向量。但是一般来说，如果原矩阵中包含过多的缺失值会导致隐含变量没有唯一解。但是这没有关系，因为我们只是想得到其中的某一个解而已。[3]

定理：隐含变量是互不相关的

在 k 近邻那一章我们讲过变量之间的多重相关问题，在选择进入模型的变量时，没有建模者希望在模型中包含进任何冗余变量。隐含变量的好处就在于它们是互不相关的。下面我们简单的证明一下：

假设我们已经找到了矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$，满足 $\boldsymbol{U} \cdot \boldsymbol{V} = \boldsymbol{X}$ 具有最小化的平方误差。这样的矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 可能有很多，我们想要的是其矩阵元素和最小的矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$。因此我们可以尝试最小化矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 内元素的平方和。不难看出，我们可以通过右乘矩阵 $\boldsymbol{U}$ 一个可逆矩阵 $\boldsymbol{G}$（$d \times d$），相应的左乘矩阵 $\boldsymbol{V}$ 一个矩阵 $\boldsymbol{G}^{-1}$ 也可以得到相同的矩阵 $\boldsymbol{X}$，因为：$\boldsymbol{U} \cdot \boldsymbol{V} = (\boldsymbol{U} \cdot \boldsymbol{G}) \cdot (\boldsymbol{G}^{-1} \cdot \boldsymbol{V}) = \boldsymbol{X}$。

假设矩阵 $\boldsymbol{G}$ 的行列式值为 1，也就是说我们在对矩阵 $\boldsymbol{U}$ 做变换的时候强制了其是体积不变的（volumn-preserving transformation）。如果我们暂时忽视矩阵 $\boldsymbol{V}$ 中元素的值，而只关心最小化矩阵 $\boldsymbol{U}$ 中元素值的大小，那么就相当于在一个 n 维空间内最小化一个 d 面体的表面积（该 d 面体的体积是固定的）。实现的唯一方法就是让该 d 面体的每一条边都相互正交，也就是说 $\boldsymbol{U}$ 矩阵内的每一列都是互不相关的。

但是不要忘记了，我们刚才似乎忽略了矩阵 $\boldsymbol{V}$！但是基于同样的道理，如果我们把注意力集中在矩阵 $\boldsymbol{V}$ 上，而选择忽略矩阵 $\boldsymbol{U}$，那么 $\boldsymbol{V}$ 的每一行必须是互不相关的。其实 SVD 从理论上已经告诉了我们 $\boldsymbol{U}$ 矩阵列之间以及 $\boldsymbol{V}$ 矩阵的行之间是相互正交的，但其实这里的 $\boldsymbol{U} \cdot \boldsymbol{V}$ 与 $\boldsymbol{X}$ 的 SVD 还是有所不同。有些人觉得 $\boldsymbol{U} \cdot \boldsymbol{V}$ 就代表着 $\boldsymbol{X}$ 的奇异值分解，其实从严格意义上来说是不对的。

现在放松矩阵的 $\boldsymbol{G}$ 的条件，允许它的行列式的值为任意值——比如说，我们允许 $\boldsymbol{G}$ 是单位矩阵的倍数形式。那么只需要一点微积分的技巧，就可以发现最佳的倍数值（最佳指的是可以最小化矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的元素平方和）是矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 所有元素的几何平均值。

这就是我们的证明过程，你被说服了吗？

8.1.8 交替最小二乘法

那么到底如何找到这样的矩阵 $\boldsymbol{U}$ 和矩阵 $\boldsymbol{V}$ 呢。从上面的叙述来看，我们似乎是先通过最小化误差平方和，再通过最小化矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的元素平方和，才能找到最佳的矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$。

注 3：从矩阵理论上来说，隐含变量解的形式与原数据矩阵中的缺失值并无关系，因此译者不能理解原书作者这段话到底想表达什么意思。

从步骤来看，是两个一先一后的步骤。但其实，这两步可以同时完成。

基本上来说，这是一个最优化的问题，但是这个最优化问题不同于普通最小二乘问题，它是没有解析解的。我们需要类似于“梯度下降法”这样的迭代算法去解决这个最优化问题。只要这个最优化的对象函数是凸函数，那么就基本可以保证算法可以最终收敛到真值。（也就意味着，算法不会在局部最优点上永久停留，而找不到全局最优点。）即便函数本身不是一个凸函数，我们也可以通过增加惩罚项的方式把函数强制地变为一个凸函数。

下面就是这个迭代算法的详细步骤。

- 随机生成一个矩阵 $\boldsymbol{V}$。
- 将矩阵 $\boldsymbol{V}$ 固定，最优化矩阵 $\boldsymbol{V}$。
- 将矩阵 $\boldsymbol{U}$ 固定，最优化矩阵 $\boldsymbol{U}$。
- 重复上述步骤知道矩阵 $\boldsymbol{U}$ 和矩阵 $\boldsymbol{V}$ 中元素的变动不再显著为止。具体来说，我们可以定义一个容忍值 ϵ，只要变动幅度不超过该容忍值，我们就认为该迭代算法已经“收敛”了。

没有证明过程的定理：如果先验信息量足够，那么刚才的迭代算法一定收敛

先验信息量越大，最优化过程会变得越简单。但是从另一方面来说，如果先验信息太多，那么所有的参数估计都会接近于 0，这对于实际问题来说没有任何意义。因此，我们从来不希望先验信息变得过于主导。但是，我们不可以选择一个最优化的先验信息量，因为如果我们这么做，先验信息就不能称为先验信息了。我们在最小化参数的过程中还要想着如何找到一个能够近似矩阵 $\boldsymbol{X}$ 的近似矩阵，这两个过程其实是南辕北辙的。你对参数的大小关注得越多，对矩阵 $\boldsymbol{X}$ 的近似就要关注得少一点。但是，我们还是认为，更多的精力应该放在如何更好地近似矩阵 $\boldsymbol{X}$ 上，这才是我们的真正目的。

8.1.9 固定矩阵$\boldsymbol{V}$，更新矩阵$\boldsymbol{U}$

在固定矩阵 $\boldsymbol{V}$，最优化矩阵 $\boldsymbol{U}$ 的那一步，最优化的过程是关于用户的。对于每一个用户 i，最优的对象是：

$$argmin_{ui} \sum_{j \in P_i} (P_{i,j} - u_i * v_j)^2$$

其中 v_j 是固定的。换句话说，这里只关心用户 i。

这看起来与最小二乘法的数学表达没有本质上的区别！的确是。根据最小二乘法的结论，上面最优化过程的最优解为：

$$u_i = (\boldsymbol{V}_{*,i}^{\tau} \boldsymbol{V}_{*,i})^{-1} \boldsymbol{V}_{*,i}^{\tau} \boldsymbol{P}_{*,i}$$

其中 $\boldsymbol{V}_{*,i}$ 是矩阵 $\boldsymbol{V}$ 的子集，其代表用户 i 对商品的喜好程度。上式中的求逆操作并不难，

因为矩阵$V_{*,i}^{\tau}V_{*,i}$是一个 d 阶方阵，是一个较小的矩阵。因为每个用户的商品喜好向量并不是一个很长的向量，因此这一步的计算量其实很小。

刚才讲到的迭代过程是固定 V 更新 U 的过程。类似地，如果固定矩阵 U 更新矩阵 V，其结论相同。求逆操作只涉及一个 d 阶方阵。

另外一个好消息是，因为用户之间的喜好是相互独立的，所以上述迭代过程完全可平行化。如果计算机的 CPU 是多核的，或者手头有几台空闲的计算机，你完全可以用平行化的方法提高算法迭代的速度。这对于大数据分析来说至关重要。

8.1.10 关于这些算法的一点思考

每一种方法都有不同的实现版本，我们已经展示了如何在一些问题上做出权衡以达到更好的预测效果。有时候，针对不同的问题也要做相应的调整，凡事还是要具体问题具体对待。

比如说，随着新用户和新数据的加入，我们可能要不断地更新矩阵 U 和矩阵 V，不断地最优化这两个矩阵。但是对于某些用户，经过细心地观察，你可能会决定他们的状态已经相对比较稳定，因此不需要再更新关于他们的那部分模型以节省计算资源。这些决定都要取决于你自己。

就像所有的机器学习模型一样，交叉验证的方法应该贯穿于建模的始终——我们要始终抽离出一部分数据用作模型的独立验证。这是有效避免过拟合问题的关键方法。

8.2 思维实验：如何过滤模型中的泡沫

对用户喜好的预测是通过最小化误差的形式实现的，这对大家有何启示呢？推荐系统的模型，从表现形式上来看会对模型的更新产生什么样的影响呢？

具体来说，商品展现给用户的先后顺序是否会影响商品本身的受欢迎程度呢？或者进一步说，商品本身是否会因为这样的人为操作而具有本身不该具有的泡沫优势呢？如果有的话，我们又如何在模型中过滤掉这些泡沫效应呢？

大家不妨静下心来好好想一想这些问题。

8.3 练习：搭建自己的推荐系统

在第 6 章中，我们有幸拿到了 GetGlue 公司提供的数据，练习了有关探索性数据分析的内容。现在，我们不妨再次利用这个数据集，搭建一个属于你自己的推荐系统。下面的 Python 代码并不是针对该数据集的，而是 Matt 为大家提供的一段示例代码。读懂这段代

码，并尝试把它应用到 GetGlue 数据集上。

Python示例代码

```
import math,numpy

pu    =    [[(0,0,1),(0,1,22),(0,2,1),(0,3,1),(0,5,0)],[(1,0,1),
(1,1,32),(1,2,0),(1,3,0),(1,4,1),(1,5,0)],[(2,0,0),(2,1,18),
(2,2,1),(2,3,1),(2,4,0),(2,5,1)],[(3,0,1),(3,1,40),(3,2,1),
(3,3,0),(3,4,0),(3,5,1)],[(4,0,0),(4,1,40),(4,2,0),(4,4,1),
(4,5,0)],[(5,0,0),(5,1,25),(5,2,1),(5,3,1),(5,4,1)]]

pv     =     [[(0,0,1),(0,1,1),(0,2,0),(0,3,1),(0,4,0),(0,5,0)],
[(1,0,22),(1,1,32),(1,2,18),(1,3,40),(1,4,40),(1,5,25)],
[(2,0,1),(2,1,0),(2,2,1),(2,3,1),(2,4,0),(2,5,1)],[(3,0,1),
(3,1,0),(3,2,1),(3,3,0),(3,5,1)],[(4,1,1),(4,2,0),(4,3,0),
(4,4,1),(4,5,1)],[(5,0,0),(5,1,0),(5,2,1),(5,3,1),(5,4,0)]]

V = numpy.mat([[0.15968384, 0.9441198 , 0.83651085],
               [ 0.73573009, 0.24906915, 0.85338239],
               [ 0.25605814, 0.6990532 , 0.50900407],
               [ 0.2405843 , 0.31848888, 0.60233653],
               [ 0.24237479, 0.15293281, 0.22240255],
               [ 0.03943766, 0.19287528, 0.95094265]])

print V

U = numpy.mat(numpy.zeros([6,3]))
L = 0.03

for iter in xrange(5):

    print "\n----- ITER %s -----"%(iter+1)

    print "U"
    urs = []
    for uset in pu:
        vo = []
        pvo = []
        for i,j,p in uset:
            vor = []
            for k in xrange(3):
                vor.append(V[j,k])
            vo.append(vor)
            pvo.append(p)
        vo = numpy.mat(vo)
        ur = numpy.linalg.inv(vo.T*vo +
            L*numpy.mat(numpy.eye(3))) *
            vo.T * numpy.mat(pvo).T
        urs.append(ur.T)
    U = numpy.vstack(urs)
    print U
```

```
    print "V"
    vrs = []
    for vset in pv:
        uo = []
        puo = []
        for j,i,p in vset:
            uor = []
            for k in xrange(3):
                uor.append(U[i,k])
            uo.append(uor)
            puo.append(p)
        uo = numpy.mat(uo)
             vr = numpy.linalg.inv(uo.T*uo + L*numpy.mat(num
py.eye(3))) * uo.T * numpy.mat(puo).T
        vrs.append(vr.T)
    V = numpy.vstack(vrs)
    print V

    err = 0.
    n = 0.
    for uset in pu:
        for i,j,p in uset:
            err += (p - (U[i]*V[j].T)[0,0])**2
            n += 1
    print math.sqrt(err/n)

print
print U*V.T
```

第 9 章

数据可视化与欺诈侦测

本章内容的贡献者是 Mark Hansen 和 Ian Wong。Mark 是哥伦比亚大学的教授，Ian 之前（2012 年 11 月我们课程进行的时候）是 Square 公司的统计学家，现在任职于 Prismatic 公司。他们两位虽然都住在加利福利亚，并且十分注重数据的可视化，但在本章他们主讲的题目将截然不同。同其他章的贡献者一样，他们两位都是深刻的思想家，他们常常思考的问题包括：什么是好的代码、编程语言作为一种人类的表达方式到底有什么样的特性。当然，也包括本书最重要的主题：到底什么是数据科学？

9.1 数据可视化的历史

Mark Hensen 来自加利福利亚大学洛杉矶分校（UCLA），在休假期间，他兼职于《纽约时报》研发实验室，并且是哥伦比亚大学布朗媒介创新研究院的院长。他博士毕业于伯克利大学统计学院，毕业之后在贝尔实验室工作了几年，随后去了 UCLA，任职于统计学部，并且同时任职于电子工程部和设计 / 媒体艺术部。Mark 在数据可视化界有相当高的知名度，同时他也是一个富有激情的演说家。某个晚上，他本来应该是在星巴克喝大杯拿铁的，却被我们拽到课堂中给大家上了一课。

Mark 会首先带我们回顾一下数据可视化的历史，最后会跟我们分享一些他的数据可视化项目。他可视化项目的成果现在都实实在在地摆在了一些博物馆或者广场上供人欣赏。他的工作哲学和理念深深地影响着他身边的人。他也在教授一些关于数据可视化的课程，他的工作往往突破常规、不拘一格，并且深深地影响着这个行业。可以说，他是数据可视化的先驱，这也是 Rachel 把他请到课堂中来的原因。在本章的最后，我们还会讲一些关于数据可视化技术前沿的内容。

9.1.1 Gabriel Tarde

Mark 首先提到了社会学家 Gabriel Tarde。Tarde 觉得社会科学产生的数据量要远远大于别的学科。他指出，其他学科对数据的态度往往倾向于隔靴搔痒，各种模型的存在也无非是对数据某种形式的变换和汇总：比如说，生物学家并不直接研究细胞，而是间接地研究细胞表现出来的某些功能。Trade 觉得这一定程度上是对信息匮乏的妥协。从生物学家的角度来看，我们真正应该研究的应该是每一个细胞本身产生的所有数据。

在社会学中也一样，每个人就是社会学中的细胞。在 Facebook 上，每个人每天都在生产大量的数据，这是研究人和社会的绝佳媒介。

但是这里自然会有一个问题：当我们太注重细节和个体的时候，很可能会失去对事物整体的把握。在社会学研究中，如果太注重个体研究，我们很容易忽略人们之间的文化属性和社交属性，而这些属性往往意义更加深刻。法国现代社会学家 Bruno Latour 在他一篇评论 Tarde 的文章“Tarde’s Idea of Quantification”（Tarde 的量化心得）中这么说过：

> **整体**，作为一个概念，可以看作其局部成分的某种方式的重构或者转化。同样的局部构件，在不同的重构和转化工具下会表现出不同形式的**整体**。
>
> ——Bruno Latour

Tarde 于 1903 年在某个类似日报的媒体上发表的一段话，可以说是对 Facebook 的一段精准的预言。他说：

> 如果统计学能够按照近几年的发展速度发展下去，如果信息的到来能够更加准确、迅捷、海量和有规律，那么一个信息化的时代终将到来。在这个时代里，**社会上每一个小事件完成后都会自动生成数据并上传到统计储存器中。这些数据将会实时、连续地通过媒体向公众传递，甚至会通过更为形象化的方式向海外传播**。到了那个时候，我们每瞄一眼报纸或者海报，都会被巨量信息所包围。**无论是社会发展的细微动态、商业竞争的即时信息、政党支持率的偏移，甚至是某种学说的兴衰，都将以统计数据来说明问题，人们对社会的认识会变得更加精简浓缩**。未来的我们，对信息的获取和处理或将成为我们本能的一部分，就像我们现在睁开眼睛就可以看到人与自然一样。在未来，信息或将成为自然的一部分。
>
> ——Tarde

Mark 于是在他的课堂中用 Brune 的一句话如是总结道：

> 改变了研究工具，就等于改变了整个社会学研究的整体面貌。
>
> ——Bruno Latour

Mark 说（Tarde 应该也这么认为），这就类似薛定谔的猫，我们应该重新审视我们观察和了解这个社会的方式，以及我们认识局部与整体关系的角度，因为当我们观察这个世界的

时候，它其实是在不停变化的，我们需要有动态的、全局的视角。

换句话说，老式的数据收集方式迫使我们使用样本和汇总统计量（比如均值）去间接地了解我们所观察的对象。但是现在呢？我们得到的数据越来越多，有时候甚至可以掌握总体的全部数据，我们自然应该改变我们的研究工具，而不应该墨守成规，总是想着用抽样和汇总的方法，我们的研究触角既可以深入到个体，也可以扩大到整体。比如我们将要在下一章中讲到的基于图论的社交网络研究，它会带给我们更多的关于研究个体本身以及研究个体之间关系的信息。这些都源于信息潮的到来，以及我们处理信息能力的增强。既然旧有的限制条件已经慢慢消失了，我们自然应该改变我们思考和研究问题的方式。

9.1.2 Mark的思维实验

随着数据的个人化特征愈发明显，更多关于人类活动本身的数据需要我们去分析。由此产生的问题就是，我们应该使用什么样的工具研究人与人、人与社会、社会与国家、国家与世界之间纷繁复杂的关系呢？

民意调查和总统支持率这样的数据分析调查框架固然可以掌握公众意见的动向，但能否展示民意中个性化和互动的部分？

即便可以，我们又是否愿意生活在这样一种环境中——我们的个性和互动信息能够被精准记录下来并随时供人取用？

9.2 到底什么是数据科学

Rachel 在第 1 章中提出并尝试回答了“到底什么是数据科学”这一问题。Mark 在这里想从一个全新的角度再一次审视这个命题。他首先提到了 John Tukey 的一句名言：

> 作为一个统计学家的最大特权就是，你可以在每一个人家的后院玩耍。
>
> ——John Tukey

如果把统计学家，抑或数据科学家，比作可以在任何人的后院玩耍的全能型人才，这样合理吗？换句话说，这样的“全能”真的是成为数据科学家的必要条件吗？到底什么样才能算作“全能”呢？

我们不妨把学科分为“数据类传统学科”和“其他学科”两类，其中前者主要包括数学、统计学、计算机科学等。“全能型”从概念上来说不过是代表对于所有的数据类传统学科的知识都谙熟于胸，因为我们不同于常人并深深地痴迷于数据科学的各个传统领域。如此看来，我们或许是过于自大了，从而导致我们看待这个问题的角度也变得非常狭隘。

因此即使是对于最简单的问题，即到底哪些学科是“传统”的数据类学科，哪些属于“其他”类，我们都还回答不上来。

在 Mark 看来，“其他”类应该包括大部分社会科学学科和理科类的学科，比如教育学、设计学、新闻学和媒体艺术学等。我们要成为的，不仅仅是单纯的“技术专家”，因为每一项技术本身其实都来自于某个研究领域对工具的实际需求。比如说，地理信息系统是由地理学家的需求衍生和发展出来的，而文本数据挖掘的技术则来自于数字人文学科的实际需求。

也就是说，每个领域的实际需求在引领着不同技术的发展。数学这样的从理论出发的学科固然重要，技术也对各个学科的发展起着至关重要的作用。当学科与数据科学交叉融合之时，都会衍生出具有学科特色的需求，进而不断地推动学科本身以及数据科学不断向前发展。

数据科学的健康发展离不开每个学科领域独立的健康发展，学科交叉和融合也变得愈发普遍和不可阻挡。在这样时代性的交叉和融合进程中，数据科学应该担当起旗手的重任，其要面对的一大难题就是什么样的交叉和融合才是健康的。

因此，Mark 在描述他自己的数据科学背景的时候指出，数据科学家是一个地地道道的“扩张主义者”，对这个概念你也许不会感到诧异。

9.2.1 Processing

Mark 在给艺术家和设计师的编程课中提到过一种叫作 Processing（http://processing.org/）的编程语言。他以 Processing 为例，展示了设计师和工程师在编程时的思维区别。一门好的数据编程语言应该以分析人员的思维方式为起点，以分析任务为导向，并且具有完整的结构性。

我们可以通过一个思维实验来进一步探讨这个问题：艺术家或者设计师到底需要什么样的编程语言？ R 在统计学家和数据科学家中流行，是因为 R 解决了他们对于某些分析任务的刚性需求，比如随机变量、分布类、向量和数据结构等。相比于 R，一门艺术家或者设计师使用的编程语言应该解决哪些刚性需求呢？

艺术家更加关心形状、轮廓等人的大脑中可以想象的图像元素，给艺术家的编程语言应该能够最大程度上以最直观的形式把人脑内的图像元素外化并且展现出来。素描、3D、动画甚至是交互的方式都可以，最重要的是能够清晰和直观地展示给人看。

Processing 就是这样一门基于 Java 的编程语言，它是专门给艺术家设计的。Mark 说他在给艺术家和设计师讲编程课的时候，要从最基本的编程技巧说起，包括迭代、`if` 语句等。因为这些基本的编程技巧在我们看来可能再平常不过，但是对于艺术家和设计师来说却是一种崭新的知识形态。因此，在给这些人上课的时候，他需要回到原点，从零开始。

9.2.2 Franco Moretti

Mark 接下来讲到了近距离和远距离文本阅读的问题。他首先提到了斯坦福大学的文学家 Fanco Moretti。

Franco 认为“远距离阅读”的效果就是不需要逐行逐句逐字地读，而只需要远远地看一下就能理解文本的大致含义。这种阅读方式有点类似于对文本内容的降维（第 8 章中我们学习了一些典型的降维技术）。

Mark 觉得 Franco 的例子很好地说明了数据科学如何在不同的学科发挥功效。与其越俎代庖，不如以一个旁观者的姿态细心学习各个学科的特点，并结合数据科学的要求将二者很好地结合起来。这样一种学习和旁观的态度对数据科学的健康发展起着方向性的作用。

9.3 一个数据可视化的方案实例

下面给大家展示一些被 Mark 所称道的可视化案例。对于每一个案例，我们都应该问这样一个问题：可视化的数据是什么？这个方案是该问题最理想的可视化方案吗？

图 9-1 是一个关于城市能源使用量的可视化项目。图中烟囱中投射出来的绿色阴影的大小代表了该城市中能源的使用量。



图 9-1：Helen Evans 和 Heiko Hanse 的 Nuage Vert 可视化案例（http://youtu.be/l_4rTQCWItw）（另见彩插图 9-1）

在另外一个叫作 One Tree 的项目中（见图 9-2），设计师 Natalie 把一个种子克隆了许多份并栽培在城市的不同区域。种子在不同的环境下成长，几年后就变成了一棵棵大树。这个案例形象地展示了环境对于植物生长的长期影响。



图 9-2：Natalie Jeremijenko 的 One Tree 可视化案例（http://boingboing.net/2003/05/16/natalie-jeremijenkos.html）

图 9-3 的案例叫作 Dusty Relief，图中的几何状建筑物可以收集其周围的污染物并以粉尘的形式表现出来。

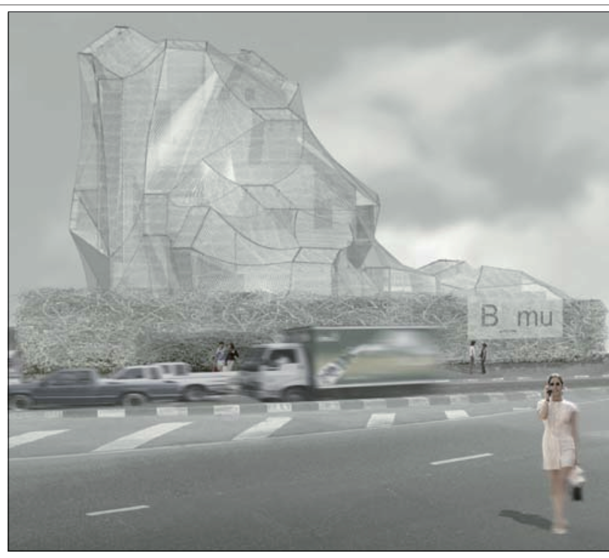


图 9-3：New Territories 的 Dusty Relief 项目（http://www.new-territories.com/roche2002bis.htm）（另见彩插图 9-3）

《纽约时报》的研发实验室开发了一个叫作 Project Reveal 的可视化项目（见图 9-4）。图中的镜子内置了一个智能的面部识别系统，只要站在镜子对面，镜子上就会显示关于你的一些个人信息。Mark 说每当站在这个镜子前面的时候，那种感觉是非常奇幻的，难以名状。

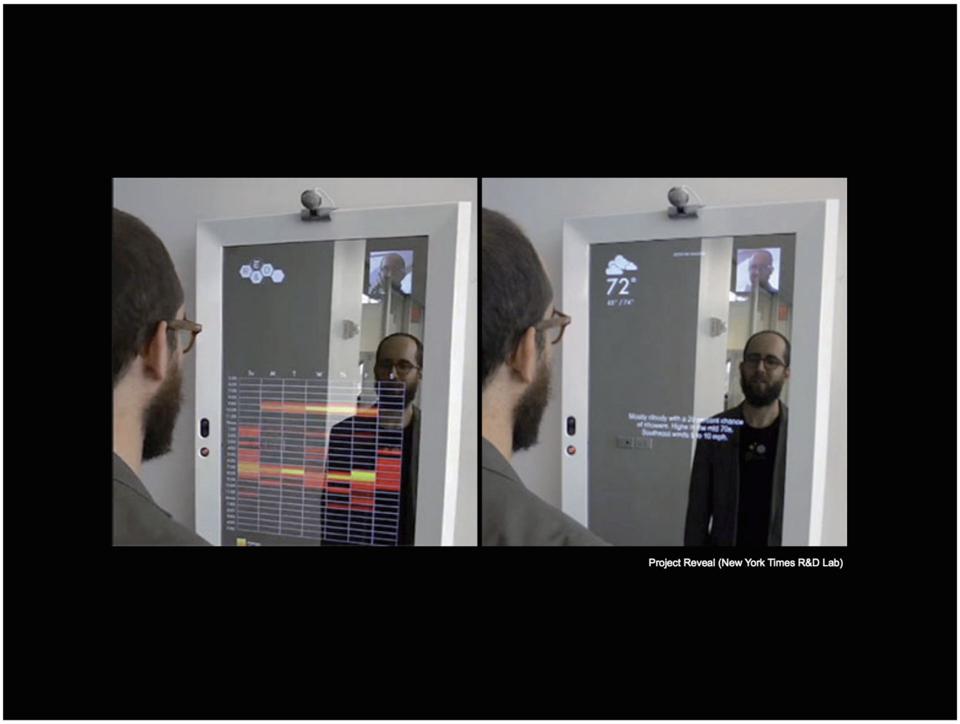


图 9-4：《纽约时报》研发实验室的 Project Reveal 项目（http://nytlabs.com/projects/mirror.html）

Laura Kurgan 是美国地理信息设计实验室（SIDL）的负责人，图 9-5 是他们设计的一个叫作“百万美元街区”的可视化项目。这个项目的数据来自谷歌的犯罪数据。Laura 提取了监狱服刑人员的家庭住址，并计算出政府管理该服刑人员所支出的费用。因此图中的颜色深度代表了政府在管理该区域的服刑人员方面所支出的费用，颜色越红代表支出越高。某些街区的费用支出甚至高于 100 万美元。这个项目的意义在于，要想在地理信息图可视化上做出新意其实是不易的。这要求研究人员的可视化思路非常开阔，对数据的理解十分透彻。像图 9-5 这样的地理信息图就体现了不一样的可视化思路。

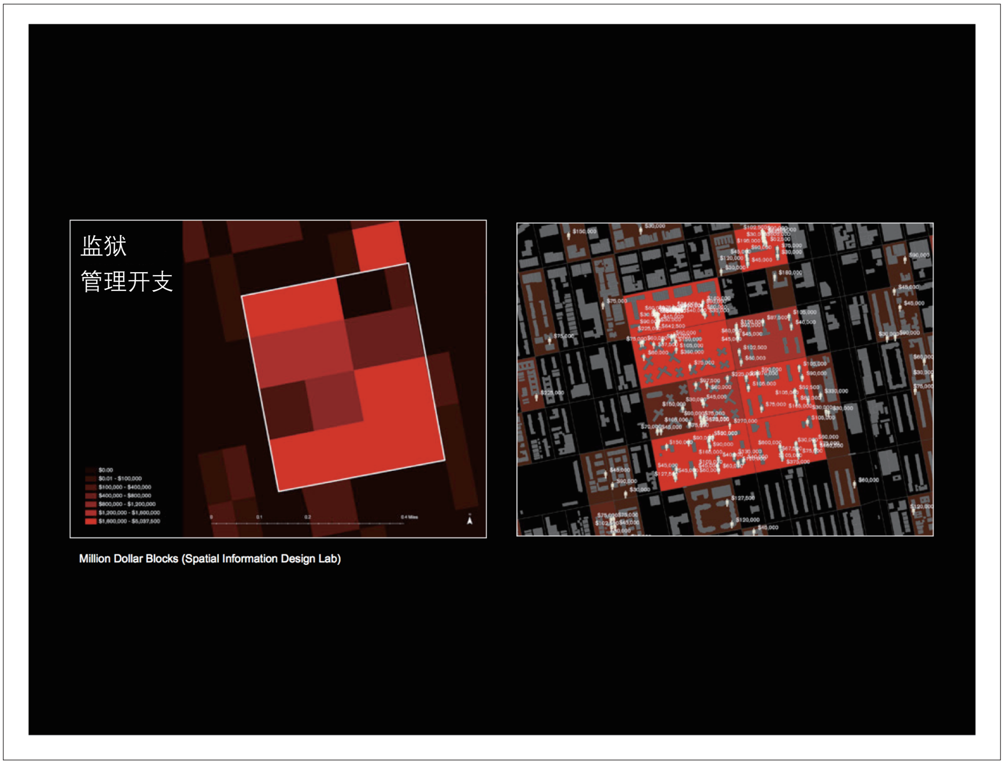


图 9-5：地理信息设计实验室（SIDL）设计的“百万美元街区”项目（http://www.spatialinformationdesignlab.org/）（另见彩插图 9-5）

9.4　Mark的数据可视化项目

我们对 Mark 的个人影响以及他的数据科学哲学已经有了大致的了解。接下来我们来看一看 Mark 本人的一些可视化项目。

9.4.1　《纽约时报》大厅里的可视化：Moveable Type

Mark 带我们参观了他和他多年的合作者媒体艺术家 Ben Rubin 一起完成的一项位于《纽约时报》总部大厅里的可视化项目（在完成这个项目之后，Mark 就利用休假的一年时间去了《纽约时报》的研发实验室工作）。《纽约时报》曼哈顿中心区总部大楼位于纽约 42 街第八大道，这个可视化项目就展示在总部大楼的一楼大厅内，图 9-6 就是这个项目在展示时的情景。



图 9-6：《纽约时报》总部大厅，Moveable Type 可视化项目，作者：Ben Rubin 和 Mark Hansen

该项目使用了两面墙，每面墙 280 个文本显示盒子，共计 560 个。每个屏幕循环显示一系列设计好的情景，每个情景都包含一个由数据模型驱动的情景主题。

对于每个显示盒子上要展示的情景，其背后的数据都来自《纽约时报》的网站内容，包括新闻、博客以及搜索引擎结果等。对网站内容的解析使用了斯坦福大学的自然语言处理技术（http://nlp.stanford.edu/），该技术可以将文本内容图解化。这些图解化的文本内容经过数据模型被进一步情景化。该项目最初设计了总计 15 个情景，每个情景的数据模型都通过编程实现，因此修改也比较容易。YouTube 上有关于 Mark 和 Ben 对于该项目的访谈视频，感兴趣的读者可以搜来看一看（http://goo.gl/yhCG69）。

举其中三个情景主题为例。一个情景是用文字波浪的形式展示对某篇文章文本的解析结果。每一篇文章都用一段简短的语句表示，不同的文章之间通过这样一段简短的语句就基本可以感知该文章的基本内容了。

另外一个情景不使用文字波浪模型，其背后的数据模型可以提取出更加简要的文本结果，通常只包含一个数值和一个名词的组合，比如“18 只大猩猩”。这样的名词组合会投射到显示器上并展示出来。

第三个情景更加有趣，显示屏会显示一个个自动化的填字游戏，其填字的过程还伴随着铅笔划纸的响声，给人的感觉就是显示屏在自己完成一个又一个填字游戏，非常生动。

图 9-7 就是一种显示盒子的内部图，看起来非常复古。每个显示屏其实都包含一个独立的、运行着 Python 的 Linux 处理器，以及一个声卡用来模拟各种各样的声音，比如钟的滴答声、打字的声音、波浪的声音等。具体发出何种声音要取决于显示屏所要展示的情景主题。

的。Jer 是《纽约时报》的数据艺术家，同时也在 bit.ly 兼职。Cascade 的可视化对象是人们在 Twitter 上对《纽约时报》文章链接的分享行为。

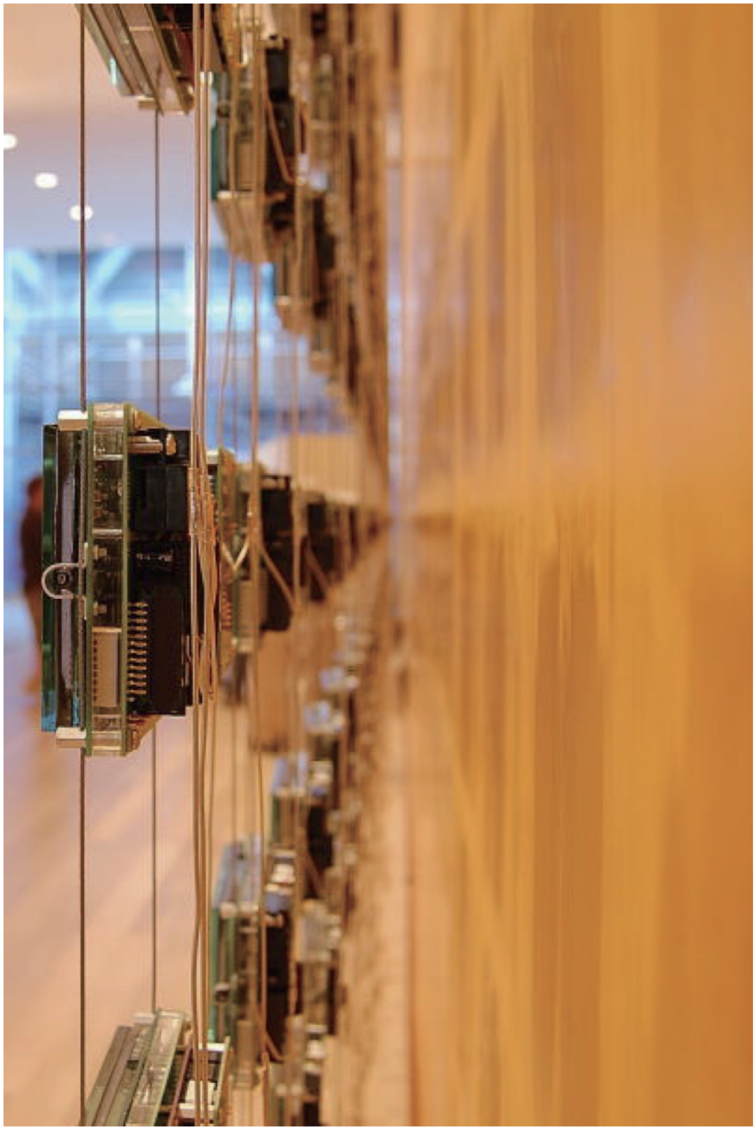


图 9-7：Moveable Type 的显示盒子

9.4.2 屏幕上的生命：Cascade可视化项目

Mark 接下来提到了 Cascade 项目，该项目是由 Mark 和他的合作者 Jer Thorp 共同设计完成的。Jer 是《纽约时报》的数据艺术家，同时也在 bit.ly 兼职。Cascade 的可视化对象是人们在 Twitter 上对《纽约时报》文章链接的分享行为。

该项目试图通过分析足够的数据，较为完整地展示《纽约时报》链接被浏览、编译（通过 bit.ly）、分享、解译（也是通过 bit.ly）和点击的过程。图 9-8 就是对这个过程的可视化结果，看起来和 Tarde 当初的建议不谋而合。

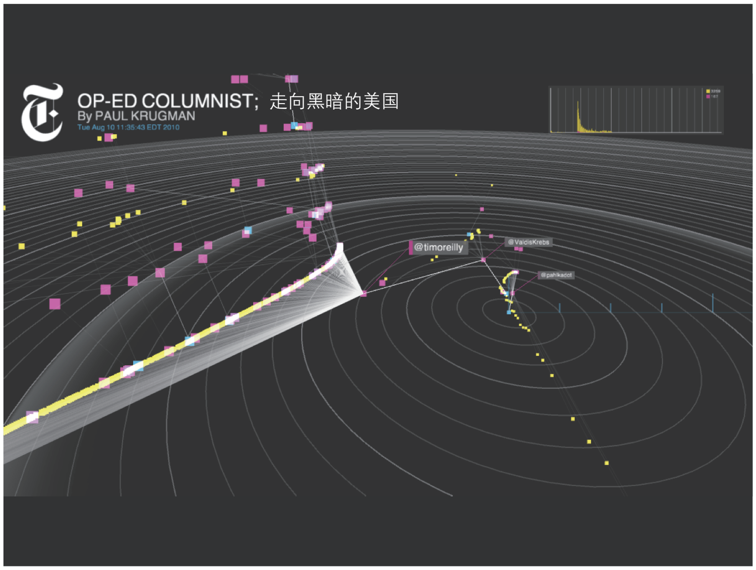


图 9-8：Jer Thorp 和 Mark Hanse 的作品：Cascade 项目（另见彩插图 9-8）

对于 Twitter 的数据可视化，比较棘手的是对数据实际意义的解读。比如说，如果有 17 条推文都是关于同一条链接的，那么就很难确认到底哪一条“推文 / 链接”组合是实际上被人点击的。我们可以通过“猜”的方式，比如使用概率模型，根据时间戳信息大致猜出哪条组合被点击的可能性最高。另外，Twitter 上的互粉关系也会被考虑进来，比如说，如果你的好友在你之前发布了同一条链接的推文，那么你的推文很可能只是对好友推文的转发。

对这个项目感兴趣的读者可以尝试去 YouTube 观看有关这个项目的视频介绍（http://goo.gl/uAOcg8）。

> 这个项目在两年前就已经完成了。在这过去的两年时间里，Twitter 的规模已经发生了翻天覆地的变化。

9.4.3 Cronkite广场项目

另外一个可视化项目展示在位于得克萨斯州大学奥斯汀分校的 Cronkite 广场。该项目由 Mark、Jer 和 Ben 共同设计完成，它仍然与新闻内容有关，但是从图 9-9 中可以看出，文字被投射在了整个大楼的外部结构上。

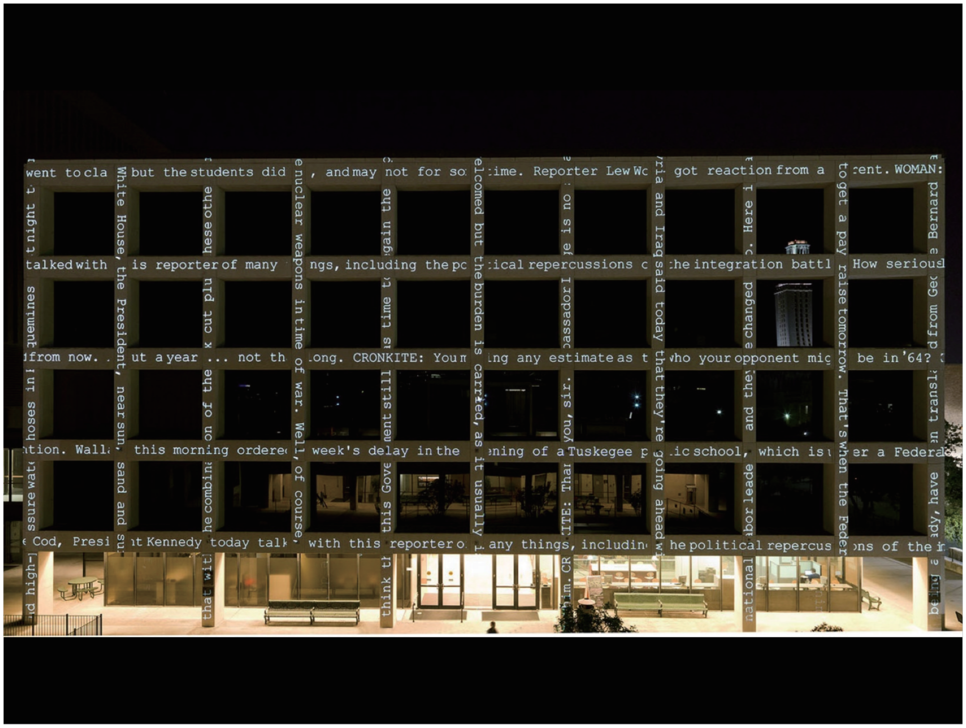


图 9-9：Jer Thorp、Mark Hanse 和 Ben Rubin 共同设计制作完成的项目：And That's The Way It Is（“就应该这样”）

在 Cronkite 广场上，每天晚上这些词语会被 6 个不同的投影仪投射在这个建筑物的外墙上。这些词语的来源包括 Walter Cronkite 的新闻联播以及一些当地的闭幕电台节目的新闻内容。词语是某条新闻内容的提炼，以句子的形式投射在外墙上。譬如，墙上会出现类似“她到底该作何反应呢？”或者“你会养什么样的狗狗？”等，类似于新闻的标题，但是更加简约。

9.4.4 eBay与图书网购

Mark 的另外一个项目是 eBay 网站上的图书网购数据的可视化，这个项目是他和 Jer Thorp 共同设计制作完成的。试想一下，如果让你把 eBay 网站上最近两年经由 PayPal 支付的图书网购数据用可视化图形的方式展现出来，你该如何下手呢？ Mark 和 Jer 为 eBay 设计了一个实体的可视化方案，并在 2012 年举办的两年一次的 ZERO1 大会上展示了他们的成果。图 9-10 就是这个项目在 eBay 大楼前展示的照片，显示的具体内容类似于图 9-11。



图 9-10：Jer Thorp 和 Mark Hanse 设计：eBay 的图书网购数据可视化项目

```
4106.00,CHINESE OLD SIGNED CERAMIC SHUDEI BONSAI POT KYUSU NR
712.50,Omron HEM-650 Wrist Blood Pressure Monitor with APS
499.99,Panasonic TC-P42X3 42" Viera Plasma HDTV
491.00,$500 Best Buy Gift Card for $491.00
372.23,Droid Incredible 2 (Verizon)
346.99,Hondo Chiquita Guitar
329.70,IMATION CD-RW 48 SPINDLES
324.25,1979 Camellias of Yunnan Souvenir Miniature Stamp sheet
287.00,DIESEL New $690 Leather Jacket Coat M
279.79,NEW CALLAWAY TOP FLITE COMPLETE MRH GOLF CLUBS SET BLUE
275.88,AMAZING LABRADORITE 925 SILVER CUFF BRACELET BA31
227.16,NEW Chefs Banquet Survival Emergency Food 330 servings
212.30,Authentic Prada Blue Bag
203.50,Superb Chinese Carved Jade Plaque 18th C.
189.00,NEW IDF Tactical Combat Vest with Removable Backpack
187.98,HP OfficeJet Pro 8500A PLUS AlO Printer   A910g    NEW
182.90,Cisco Small Business RV042 Dual WAN VPN Router -  RV042
174.95,PHILIPPE STARCK Black VEILED Stainless VEIL NEW! PH5022
173.23,14K. GOLD CHANDELIERS EARRINGS NATURAL BLUE TOPAZ GEMS
137.99,FUJI FINEPIX JZ300 10x 12MP DIGITAL CAMERA SILVER NEW
126.00,New Stock White/Ivory Wedding Dress Sz:6/8/10/12/14/16
124.64,Lot of 700 HP 02 Mixed Colors Empty Ink Tank Cartridges
116.40,Goddess of Wisdom 2001 Barbie Doll NIB Mint Condition
112.08,Littmann Cardiology III Stethescope
111.11,ALLEN EDMONDS PARK AVENUE OX BLACK 8 D MEDIUM $325
105.95,REPRODUCTION GERMAN WWII WOOL SOCKS SIZE 2 RING (9-10)
99.00,TOSHIBA SATELLITE L305-S5875 15.4" WXGA  LCD SCREEN
97.58,calvin klein men's slim fit plaid suit pants
90.00,Pear Shape Diamond
80.30,HAPPY CHLOE DUCK 2010 SWAROVSKI  RETIRED  #1041293
79.99,Cisco 2600 series 2611 2-port Ethernet Router CCNA CCNP
77.77,PRO BICYCLE MECHANICS XLC TOOL KIT 33pc BIKE REPAIR SET
74.90,NEW LG VX10000 VOYAGER QWERTY TXT MP3 PHONE VERIZON
69.94,Modern Jacquard Bedding Comforter Set Queen Black/Grey
67.49,pearl gameboy advanced SP metal case 18 gamesECT.
```

图 9-11：eBay 项目背后的真实数据

他们最后想出来的可视化方案非常具有独创性。首先，他们选了 Arthur Miller 的著作《推销员之死》（http://en.wikipedia.org/wiki/Death_of_a_Salesman）里的文本内容，并使用土耳其机器人（见第 7 章）在文本中选择可以在 eBay 上买到的东西，比如“椅子”“长笛”“桌子”等类似的商品名。

当收集到一串（10 个左右）商品名后，再在当天的销售额数据中找出该商品的交易情况，并找到一些异常值，比如最大值和最小值。在检查完每一个商品名的销售情况之后，程序会在美国的邮编数据中挑选一个邮编号码，通常是一些小地方，比如蒙大拿州。

接下来书籍的网购数据被定位到蒙大拿州，在所有来自蒙大拿州的支付信息中随机挑选一本书，再从这本书里找可以买到的商品名，并重复上述过程。要直观地了解整个可视化过程是如何进行的，可以参考网上提供的一段视频（https://vimeo.com/50146828）。

9.4.5　公共剧场里的“莎士比亚机”

最后一件作品来自于 Mark、Rubin 和 Thorp，安装在美国公共剧院的大厅里。该项目是由 37 块 LED 片形灯组构成的一个穹顶型结构，被吊顶安装在大厅接待处的正上方，具体可参考图 9-12。每一块片形灯组都对应于莎士比亚 37 部戏剧中的一部，灯组的长度越长代表该剧的长度越长。

因此，当你已进入大厅的时候，正对着你的就是莎士比亚作品中最长的一部戏剧作品《哈姆雷特》。



图 9-12：“莎士比亚机”，作者：Mark、Jer 和 Ben

每片灯组负责展示相应戏剧的不同主题，输入的数据就是该剧所有的文本内容，而主题不同显示的内容也不同。比如说，某个主题只显示该剧正文中出现的某个名词，某个主题选择显示词组，而另外一个主题可能会选择显示一些剧中出现过的组合词。

注意，针对这些数据，这里的所谓数字人文通过 MONK 项目（http://monkproject.org）提供了极为丰富的 XML（http://en.wikipedia.org/wiki/XML）展示。

Mark 说，由于是莎士比亚的缘故，因此不管用什么样的主题，这样的可视化方式都会显得很棒。但是，他们也在考虑如何通过这样的灯组更好地展现每部戏剧的特色：比如把字符看作符号，或者看作关键词或者演讲词等，再通过合适的组合方式更加多元化地展示每一部戏剧。

现在让我们回到 Mark 在最初向大家提出的一个最基本的问题：到底什么是数据呢？答案是：数据就是生活，它无处不在。

Mark 在最后对如何获取数据给了一条很中肯的建议：做一个谦卑有礼的数据搜集者，因为人们总是更愿意将数据交给他们觉得更谦卑和有礼的人。

9.4.6 这些展览的目的是什么

这些展览既具有艺术和美学特征，又内容丰富，并且在设计时也避免了过重的说教性质。总体的设计原则是在美学和内容间找到一个平衡点。这些展览都具有故事性，不会让观众觉得乏味。由于各种编程和设计工具的高度整合和数字化，统计学家的作品看起来也越来越具有艺术性，设计师的作品也开始包含更多的数据和统计学元素。这样的整合趋势在刚才所展现的很多展览作品中，大家应该都深有感触。

9.5 数据科学和风险

接下来来自旧金山的 Ian Wong 将带来有关数据科学和风险管理的内容。Ian 是 Square 公司的统计学家，他博士就读于斯坦福大学电子工程专业，主攻机器学习方向，但是他中途退学了（他获得过统计学和电子工程学的硕士学位）。在讲课的时候，Ian 尚就职于 Square，但他随后就从 Square 跳槽去了 Prismatic，一家个性化新闻订阅服务商。

Ian 在刚开始就给了大家三条心得。

- 机器学习不等于写R程序

  机器学习根植于数学，它的最终成果是通过编程实现的软件产品的有机组合。要做好机器学习，你必须具备相当的程序开发技能，能写一手漂亮的程序，且这些程序要具备良好的可读性和可实现性。程序的最终受众是广大用户而不是你自己，因此你应该确保程序经得起反复推敲，并能够实现产品化。

- 数据可视化不仅仅是一张好图

  对于一家优秀的数据产品公司来说，可视化应该成为产品开发文化的一个重要组成部分。好的数据产品应该有经过仔细构思的可视化模块，同时，好的可视化模块能够带给用户更好的产品体验。

- 机器学习与可视化推动着人工智能的发展

  人类本身的认知能力是十分有限的。然而，借助于数据和数据科学，我们有如披上了超人的外衣，得以在原本一片混沌的信息世界中自由翱翔。

9.5.1 关于Square公司

Square 成立于 2009 年，其创始人是 Jack Dorsey 和 Jim Mckelvey。公司在 2011 年有 50 名员工，到 2013 年已经发展到了 500 名员工。

Square 的创始人觉得，现实生活中的交易都太过复杂，无论从支付方还是收款方来看，现有的支付方式似乎都显得过于烦琐。Square 要解决的核心问题就是，如何使交易变得简单易行，以至于每个人都可以轻松地参与到商业活动中来？

Square 给出了一种商业模式。首先收款方（商家）需要在 Square 上成为注册用户，下载 Square 开发的官方程序，之后 Square 公司会将一个信用卡读卡器邮寄给该商家。商家在收到读卡器之后，可以把读卡器和手机绑定（例如将读卡器插在手机的数据线接口上），打开 Square 程序即可接受支付。Square 提供的小型读卡器为很多小型商家提供了一种最为迅捷的支付方式。在波特兰和旧金山，很多赶时髦的咖啡馆都开始采用 Square 的支付方式。对于消费者来说，所要做的只是拿出信用卡，在 Square 的读卡器上刷卡，最后在 iPad 上签上自己的名字。

当然，如果消费者（买方）成为 Square 的注册用户，也可以从中受益。你可以使用 Square 提供的电子钱包程序，这样当你在付款给 Square 的注册卖家时，甚至都不需要出示信用卡便可以直接支付，卖家所要做的只是在 iPad 的 Square 程序里点击你的名字。

固然，Square 想为卖家提供最为迅捷的支付服务，但他们也同样需要做好电子支付的风险防范工作。下面我们就详细讨论一下 Square 支付模式所面临的风险以及相应的防范措施。

9.5.2 支付风险

Square 的宗旨是为买方和卖方提供最为迅捷和自由的支付体验，然而这促生了一小批可能会滥用该服务的使用者。比如说，支付欺诈这样的违法行为不仅会严重损害用户体验，也是在挑战公司的价值底线。因此，如何打造一个稳健并且高度有效的风险管理系统是 Square 公司想要保持健康发展所要解决的核心问题。

那么到底如何才能有效地检测用户的支付操作并侦探出可能的欺诈行为呢？Ian 解释说，他们在开发这样一套风险管理系统时，很好地将机器学习和数据可视化技术结合了起来。

机器学习在可疑支付行为侦测中的应用

到底什么样的支付行为是“可疑”的？这是我们要最先定义的概念。以下一些异常现象要引起我们的注意：同一时间内进行了大量小额支付、突然发生的大额交易或者是不同于平常交易频率的交易（不一致交易频率）等。

我们这里给出一个例子。比如 John 有一辆运送食材的卡车，在他开始物流业务的几周后，他在 Square 上的第一笔支付金额是 1000 美元。那么这样的一笔支付有没有可能是可疑支付呢？这很难直接看出来，因为 John 可能是有偿还能力的（可靠用户）。然而一旦 John 是一个支付欺诈者，那么所有的支付金额都将由 Square 自己买单。因此，Square 所能赚取的利润与他们的风险管理水平有着直接关系，他们应该想尽办法尽量避免不可靠用户使用他们的支付服务。

但是反过来说，如果 John 有足够的偿付能力，而 Square 将他视作不可靠用户，这会在很大程度上影响 Square 的公司声誉和产品的用户体验，也会直接影响 Square 的利润率。因此，Square 的风险管理系统既不能过于保守，也不能过于宽松；过于保守会失去很多可能的可靠用户，而过于宽松会让很多欺诈用户有机可乘。

这个例子很好地说明了 Square 公司在开发风险管理系统时所面临的两难境地，从机器学习的角度来说，Square 需要同时考虑伪阳性和伪阴性的问题：伪阳性会损害用户体验，而伪阴性会让公司蒙受直接的经济损失。

从公司的角度来看，来自于买方和卖方的风险都需要考虑。本章主要考虑来自于卖方的风险，也就是收款方（比如 John）可能带来的风险。

Square 公司每天要处理的交易额高达数百万美元，因此这样一个风险管理系统需要自动化，并且有足够快的反应速度。对每一笔交易和每一个用户，都能及时做出判断。那么这样一个系统到底应该如何实现呢？其背后的数据又是什么样子呢？ Ian 先展示了该系统中的三种数据形态，可见图 9-13、图 9-14 和图 9-15。

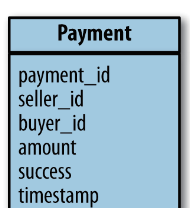


图 9-13：支付数据

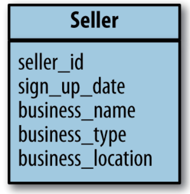


图 9-14：卖方数据

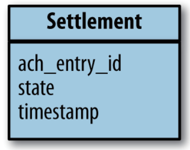


图 9-15：结算数据

这里一共涉及三种类型的数据：

- 第一种是支付数据，包含交易号、卖家识别号、买家识别号、交易额、交易成功与否、以及时间戳等信息；
- 第二种是卖家数据，包括卖家识别号、卖家注册日期、卖家店铺名称、店铺类型、店铺位置等信息；
- 第三种是结算数据，包括结算识别号、结算发生地和时间戳信息等。

在 Square 上的每笔交易都会在一整天之后结算，因此风险管理系统对交易的风险度识别并不需要在分秒内完成。当然，处理的速度固然是越快越好。

图 9-16 是这个风险管理系统的交易处理流程图。每一项交易活动的具体信息都会经过风险模型的分析，完全不可疑的交易可以直接通过审查，可疑的交易会再经过一遍人工核查。人工核查会更加仔细，分析人员会就可疑的交易信息逐条分析以确保风险可控。在人工核查阶段确认可疑的交易会被冻结，该项交易会交给后续分析人员以作进一步的深度核查。所有确认安全的交易会进入结算阶段（图 9-15 的结算数据就来自于该阶段）。

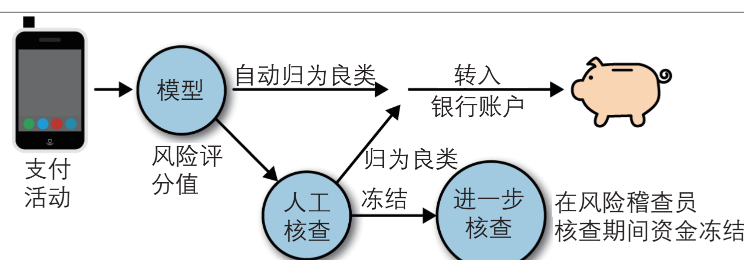


图 9-16：风险管理模型的运行流程图

接下来的问题就是如何搭建图 9-16 中的风险模型。同所有机器学习模型一样，风险模型的输入值是有关交易的特征变量，输出值可以是一个二元值（1 代表可疑，0 代表不可疑）。因此从机器学习的角度来看，这是一个典型的监督式学习问题。但从操作层面来看，这个模型远远不止一个监督学习模型那么简单。从流程图中我们知道，模型的结果要比单纯的二元值复杂得多，有些交易会被标记为良性交易而直接通过，有些会被标记为恶性交易而直接否决，更多的是可疑交易，需要交给人工核查人员以便进一步确认其风险。具体有多少种标记方式要由风险管理团队讨论决定。

因此，严格意义上来说，这算是一个半监督学习模型，它具备监督学习模型的典型特征，也有一定程度的非监督学习特性。因为非监督模型的特性，到底应该有多少个标签是需要不断调整和改变的。但是需要注意的是，在经过几个月的模型打磨之后，模型的非监督性质会逐渐消失，最后会成为一个严格意义上的监督性学习模型。Ian 接下来要着重与大家分享的内容主要是有关监督性学习模型的。

众多周知，监督性学习模型大致包括以下几个步骤：

- 获取数据；
- 提取特征变量；
- 训练模型；
- 评估模型效果；
- 应用模型。

然而，在实际问题中合理地实现这些步骤也并非易事。就连上述步骤的顺序是否应该严格遵守都众说纷纭。Ian 则建议大家在建模之前应该充分地分析问题，了解问题的实质并且明确分析的目标。也就是说，要在模型建立之前就想好如何评估模型的效果。

9.5.3 模型效果的评估问题

现在让我们集中精力关注一下模型效果的评估问题。Ian 提到，人们大约会在以下三种情况下遇到麻烦。

定义误差指标

到底如何衡量模型的建模效果是十分重要的问题。之前我们讨论过的混淆矩阵，又称作真值功能表，是衡量分类模型效果的标准工具，如表 9-1 所示。

表9-1：实际值预测值列联表，也叫作混淆矩阵

	实际值 = 真	实际值 = 假
预测值 = 真	TP（真阳性）	FP（伪阳性）
预测值 = 假	FN（伪阴性）	TN（真阴性）

最常见的模型指标是模型的准确度（accuracy），根据表 9-1 中的符号表示，它的定义为：

$$\text{准确度} = \frac{TP + TN}{TP + TN + FP + FN}$$

准确度可以解释为模型答对标签类型的概率。但是对于交易中的欺诈行为来说，因为阳性值的总数偏少[1]，因此准确度很难准确地衡量模型预测效果的好坏。这是因为即便假设所有的预测值都是阴性，模型也会有较高的准确度，因为大多数实际值都是阴性。但是很明显这个假设没有任何的预测价值，我们所关心的是阳性值。

在这种情况下，精确度（precision）或者召回率（recall）则能更好地体现模型的预测效果。精确度的定义如下：

$$\text{精确度} = \frac{TP}{TP + FP}$$

它的含义是指被预测为欺诈的交易事实上确实是欺诈交易的概率。

召回率的定义为：

$$\text{召回率} = \frac{TP}{TP + FN}$$

它的含义是事实上是欺诈的交易可以被模型侦测出来的概率。到底应该使用哪一个指标取决于很多因素。就成本因素来说，未侦测到的欺诈行为与侦测有误的交易所导致的成本，对于不同的数据和不同的数据问题，其区别是巨大的，而且前者的计算比较容易，后者属于潜在成本，很难准确计算。

定义标签

标签值对于监督性学习的重要性不言而喻，Ian 认为标签信息是“被忽视的另一半数据”。在本科阶段的统计学课程和很多数据科学竞赛中，通常数据的标签信息都是给定的。但在实际数据中，标签信息是最难定义和捕捉的信息，但对于建模来说又是最为重要的信息。它不仅与我们的目标函数有关，它在很多情况下就是我们建模的目的所在。

对 Square 公司来说，定义好标签值是至关重要的一环，因为准确地定义标签值意味着准确地定义了以下概念。

- 什么是可疑的交易。
- 数据的“粒度”如何？是以事件（交易）为主体，还是以个体（用户）为主体，还是以两者为主体？
- 我们能否得到可靠的标签值？是否需要从外部数据中获取更多的信息以帮助做出判断？

注 1：也就是说大多数交易还是属于正常交易。

最后 Ian 提到，标签值定义中的不确定性将给模型本身的预测精度带来很大影响，其中一种情况就是标签值的“高度失调”（比如之前提到的阳性数据量过少的情况）。

特征选择与模型学习过程中的诸多挑战

Ian 说，特征变量的提取是对领域知识的抽象化和模型化。一旦机器学习模型搭建好并开始运行之后，基本上大量的精力会放在如何更好地抽象化领域知识、提取特征变量上：比如，根据模型的反馈，添加新的特征变量。然而，新变量也并非越多越好，因为每个变量的可学习程度是有限的。

当数据明显失衡的时候，要千万小心“过拟合”的出现。充分地学习好一个特征变量所要求的样本量，与模型所要预测的标签值数量是成正比的（在欺诈侦测的例子中，就是欺诈交易的样本量）。

当某个变量是分类型变量并包含较多类别时，其在模型中的应用要引起分析人员的额外注意。比如说“邮政编码”这样的分类变量可能有成百上千个不同的类别，但是在建模过程中，这样多类别的分类变量基本是无用的。需要通过组合的方式加以简化。Ian 和他的团队甚至动用了模型，只为了减少某些分类变量的类别个数。

第二个问题与数据的稀疏度有关。对于新注册的卖家，这个问题显得尤为重要。新卖家的注册信息对于模型来说是未知的，而在建模之前又必须先做好特征的定义和提取。如果新卖家的注册内容包含很多新的信息，已有的特征指标系统就会显得过于保守和陈旧，不能很好地纳入新的信息。

最后的问题，也是机器学习中最头疼的问题之一：你需要根据反馈不断地调整模型，直到模型达到满意的预测效果。模型调整的范围十分宽泛，找到一个最佳模型无异于大海捞针。比如说，你需要决定是否考虑特征变量间的交互性，是否就“类别失衡”问题给予相应的调整；如果使用的是集成模型，你还需要确定一个合适的抽样框架。

另外一个棘手的问题是用户的刻意对抗行为。在电子商务中，一个恶意的用户会想方设法规避风险系统的欺诈侦测。一种典型的做法是，用户会注册十多个账户，每个账户所用的注册地址都有细微的差别，如果侦测系统主要在用户的注册地址上做文章，那么这样的做法对于逃避侦测系统的检测会十分有效。对于风险管理系统的设计者来说，要严加防范用户的对抗行为。并且，由于这样对抗用户一直在学习如何规避系统的检测，因此系统的设计和管理人员也要不停地更新系统的核心模型。

关于变量的标签值

在加州大学伯克利分校信息学院举办的一年一度的 DataEDGE 会议上，Micheal Chui（任职于麦肯锡）与 Itamar Rosenn（Facebook 聘请的第一位数据科学家）讨论了如何更好地定义“重度用户”。这其实直接涉及标签的定义问题。判断一个用户是否是“重度用户”对于企业来说十分重要。但是什么样的用户可以定义为“重度”却没有统一的标准。事实上也不需要统一的标准，因为不同的公司在不同的数据问题上，对“重度”的定义可以不尽相同。但大体上，“重度用户”具有一些共有的行为特征，譬如他们访问服务的频率和次数很高，他们创建和更新日志的次数较多等。从某种意义上来说，这是一个半监督学习问题，因为标签值的定义和预测过程是在同时进行的。

9.5.4　建模小贴士

下面是一些有关如何更好地打造模型产品的贴士。

- 模型不同于黑匣子

  模型产品最好具有高度的可解释性，我们不能假设使用的模型是万能的，因此当模型对数据失效的时候，我们可以打开模型，看看里面到底发生了什么，以便更好地改进模型。

- 勇于尝试

  做模型和做科学实验是类似的，你需要不停地尝试，模型不会一蹴而就，也不会一劳永逸。当你觉得模型 *A* 和模型 *B* 都合理的时候，应该毫无犹豫地把两个模型都尝试一遍。当然，模型的尝试也有一个上限，你不能总是处在尝试阶段，产品总是要拿出来的。

- 模型和软件包不是万能的

  你也许经常会听到人们讨论“你用的是哪个模型”“你用哪个软件包”等，这样的人往往不知道他们在做什么。真正懂模型的人应该关注模型内部的细节，而不是模型本身，或者是使用什么样的工具。

Ian 同样也注意到，人们就某种算法经常会讨论使用哪个软件包的问题。比如，如果一群人都在用 R 做机器学习，他们可能在讨论 `randomForest`、`gbm`、`glmnet`、`caret`、`ggplot2`、`rocr` 等软件包。如果是一群用 `scikit-learn`（Python 的机器学习模块）的人，他们或许会讨论 `RandomForestClassifier` 或者 `RidgeClassifier` 等。但是,这样的讨论其实没有必要，软件包毕竟只是工具，真正重要的是模型背后的理论。

程序的可用性与可读性

对于数据科学家来说，编程是一项必备技能。对于写程序，Ian 鼓励大家要注重程序的正确性、结构性、可用性和可读性。

如果你不使用别人编写好的软件包，而是自己写算法，那么代码的可读性和可扩展性会直接影响之后的产品质量。举个例子，在编写一个随机森林算法时，如果硬编码随机森林两大参数之中的任何一个，都会导致程序的后期扩展相当困难。如果有可能，参数要用软编码，这样不仅用户可以改变它的取值，后期的扩展和接口也会相对轻松很多。另外，不管你有多忙，时间有多赶，一定要写好测试程序。

在 Square，为了保证程序的可用性和可读性，编程人员根据机器学习模型的特征将程序按照语义划分为不同的构件并存放在不同的文件夹中。大体包括以下 5 个构件。

- 模型
  核心算法部分

- 信号
  数据读取和变量生成

- 误差
  模型效果评估

- 实验
  一些探索性数据分析和实验等

- 测试
  所有的测试用代码

当项目上线之后，编程工作基本会集中在“实验”文件夹内。大量的新数据会带来新的挑战，模型需要不断地改进，因此分析人员要不断地实验，做探索性分析，找到模型新的改进点。 这样做不仅能促使探索性分析带给分析人员新思维和新视角，也能尽量避免重复性劳动。对于团队来说，你可以看到同事都尝试了哪些分析，得到了哪些结论，这样在自己探索时，就可以“站在同事的肩膀上”了。Ian 直言不讳地说道，在团队中工作，就是要写生产型的代码（production code）。

项目应该包含哪些构件和文件夹并没有成文规定，John Myles White 对此有过讨论，可见这个 Project Template 项目（http://projecttemplate.net/）。如果你是 R 用户，Ian 建议多去 Github 上看一看大牛的 R 项目的代码。如果有可能，你也可以尝试写一些软件包。Hadley Wickham 的 devtools（http://adv-r.had.co.nz/）包是广大 R 包开发者的福音。Ian 说，要想写出漂亮的代码，与写出漂亮的公式证明十分相似：你需要大量认真的练习，从失败和用户的反馈中不断学习和改进。

作为课后练习，Ian 建议大家比较一下 R 中的 caret 包（http://cran.r-project.org/web/packages/caret/index.html）和 Python 中的 scikit-learn 模块（https://github.com/scikit-learn/scikit-learn），看一看谁的源代码具有更好的扩展性和可读性？为什么？

找到小伙伴

编程是一门艺术，但也讲求团队协作。闭门造车往往事倍功半。就好比你学一门外语而没有人跟你交流一样，你永远也无法掌握这门语言的精髓。

在写程序的时候找到一个志同道合的小伙伴很重要，因为互相可以检查代码、发现问题并及时交流和改正。在 Square，每一位工程师都有义务检查其他工程师写的代码。这不仅仅是为了改正代码中可能存在的错误，同时也保证了代码的质量，这包括代码的可用性和可读性等。

当你找到自己的编程小伙伴之后，还要找到一个你们共同感兴趣的问题，这样才能一起编程。买一套工作站并配上两套鼠标和键盘，接下来就可以一起协作了。你们会互相讨论如何解决面临的问题，将问题解析成块并一一攻破。这就好比汽车比赛，一个做驾驶员一个做副驾驶。驾驶员是真正操控车辆的人，也就是真正写代码的那个人，而副驾驶则负责检查代码并指示未来前进的方向。当驾驶员在写代码的时候，副驾驶应该不断地问自己“我能否看懂他写的这段代码”，或者“这段代码能够写得更简洁一些吗”等类似的问题。当副驾驶犯糊涂的时候，应该及时与驾驶员沟通以消除疑惑。找到小伙伴的目的就是为了相互监督和学习，在学习中成长。

当然，驾驶员和副驾驶应该定期交换角色。如果你们合作得很默契，你会发现一切都变得十分高效。但是协作其实是一件费神费力的工作，刚开始人会很容易疲惫。但是随着默契度的增加和协作模式的完善，连续工作的时间会越来越长。

如果身边没有这样的小伙伴，你还可以使用 git。熟悉 git 的工作流程，在拉拽请求（pull request）时要有建设性的意见和贡献。这和学术界的“同行评审”是一个道理。

将机器学习模型产品化

机器学习模型产品化的过程中有很多非常棘手的问题，如下。

(1) 到底如何将一个模型“产品化”？
(2) 如何实时地将实际数据转换成“特征变量”，并传递给模型？
(3) 如何遵守“所见即所得”的原则？也就是说，如何最小化产品在线上和线下的表现差距？

在课堂上以及大多数机器学习竞赛中，模型的评估数据是静态的，对于模型的运行速度及模型运行的实时性没有要求。因此，人们总是倾向于使用复杂的模型，因为能够得到更好的预测效果。这导致大多数模型都非常臃肿，使用的特征变量都极尽复杂。很多数据科学家往往对此不以为意，因为他们认为似乎凡事都有解决办法，如下。

- 数据的维度太高？没关系！用奇异值分解就完事了。
- 数据是有关到达率的？没关系，让我们先对历史数据拟合一个泊松模型即可。
- 时间序列数据？用傅里叶变换吧！

然而，由于存在种种时间或者空间上的限制，我们在建模的时候并不能随心所欲。从产品的角度来说，模型需在瞬间做出反应并给出预测值。因此，要尽量避免模型被不必要地复杂化。

很多模型从理论上来看，其核心部分无非是特征变量之间的点积（比如广义线性模型和支持向量机模型等），或者是一系列有关特征变量的 `if-else` 关系语句（决策树模型）。因此，模型中最耗费时间的部分往往是特征变量的种种计算。计算方式有两种方式：批量式和实时式。

由于模型的特征变量规模、复杂程度以及时延要求的不同，特征变量的计算方式也有所不同：可以采取纯粹的批量式或者实时式，也可以两种方式结合。具体采用何种方式，要具体问题具体处理。需要实时预测的情况，最好在模型的训练阶段采取批量式的计算方式，在预测阶段则采取实时式的计算。

MapReduce 是一种常用的批量式计算框架。但由于 Square 苛刻的时延要求，Ian 和他的机器学习团队在尝试开发一个实时的特征变量计算系统。

该系统的设计要保证历史特征变量和在线特征的计算方式绝对一致。换句话说，不论变量来自于训练阶段还是实时数据，都应使用相同的计算方式。只有这样，建模者才能确信模型可以正常工作。

9.6 数据可视化在Square

Ian 接下来列举了 Square 的风险管理团队应用数据可视化完成的一些任务：

- 使交易的审核更加有效；
- 更好地展示单个客户的行为模式以及客户间的行为模式；
- 监测商业运作的健康度；
- 产品环境分析。

他们开发了一个检视用户行为的工作流程系统，该系统实时地展示用户的特征，包括用户的历史交易行为、地理位置信息、用户评价、联系方式等。该流程系统在很大程度上依赖于数据可视化技术。营运团队每天要处理大量的用户交易检视任务，如果没有软件工程师和数据科学家提供类似工作流程系统的帮助，他们每天加班到凌晨也无法完成任务。有了这样的可视化流程系统，营运团队的工作会变得十分轻松，可疑的交易行为和用户的侦测会变得更加容易。Ian 觉得，机器学习和数据可视化的有效结合，能够碰撞出很多思维和技术的火花，这是一个数据公司所求之不得的。

Square 的办公室里安装了很多电视机，上面显示着公司数以千万计的客户们的实时状态。大家亲切地称呼这些电视机为“信息发射机”。这种可视化与欺诈侦测并没有直接的关系，

却能够把公司运行的实时趋势和状态展现给员工，这可以极大地激发大家的工作热情。

这样的可视化与“产品环境分析”的概念很像，它直观地提供了与产品间接相关的因素的运行状态，这对开发产品本身是大有裨益的。它有助于我们对于产品直接相关的数据的认识更加透彻和全方位。有些产品环境甚至可以重点可视化，Square 的风险管理团队就针对很多产品环境因素开发了单独的可视化界面。

Ian 的团队甚至可视化了一批风险管理指标，其中包括可疑交易每天的“清理率”和“冻结率”、每天处理的可疑交易数等。整个指标可视化系统看起来十分高端，Ian 甚至可以直接看出哪些分析人员处理得可疑案例最多、平均每个账户的处理时间是多少、哪些因素会增加核查的负担等。

总而言之，Square 是可视化技术的忠实拥趸，用户的大部分行为都被纳入到了可视化管理系统当中。可视化分析俨然成为了 Square 公司“产品环境分析”的重要一环，发挥着重要的作用。尤其对于 Ian 的风险管理团队来说，他们尤其相信，“看得清才能行得远”。

Ian 最后讲到了自己的数据科学背景以及从事这个行业的一些建议。首先，他形象地说，如果要画出自己的技能水平与数据科学背景的关系图可以用 R 函数 `plot(skill_level ~ attributes | ian)`，那么技能水平应该取对数。这是因为学习一项技能通常并不需要太久的时间，但是要精通该项技能却需要长时间坚持不懈地练习。 他同样认为生产力的测度同样应该取对数，对于那些引领同行的软件开发者们来说，它们开发一个软件包的速度要远远快于其他开发者。

作为临别赠言，Ian 鼓励大家：

- 要玩就玩真实的数据；
- 在学校里好好地学数学、统计学和计算机；
- 多去公司实习；
- 保持一颗好奇的心。

9.7 Ian的思维实验

假设现实生活发生的每条交易的数据都可以被记录和保存下来，你应该怎么使用这样的数据？

9.8 关于数据可视化

并不是每个人都能做出类似 Mark 所展示的那些可视化项目，但是掌握一些可视化技术对提高我们的数据分析技术以及交流能力会有如虎添翼的作用。同数据科学一样，数据可视化本身也是一样综合性的技术，要想真正掌握并成为大师绝非易事。为了方便大学进一步

学习可视化技术，下面列出一些我们认为比较经典的书籍或者资料。

- 关于数据可视化的一些基石性的技术，Micheal Dubokov 的介绍很不错，可参考 http://www.targetprocess.com/articles/visual-encoding.html。
- Nathan Yau 是 Mark Hanse 在 UCLA 的博士研究生。他在 http://flowingdata.com/ 发表了很多关于 R 中可视化技术的博客和文章。Nathan 也写过两本有关数据可视化的书：第一本叫作 *Visulize This: The Flowing Data Guide to Design, Visualization, and Statistics*，由 Wiley 出版社出版；另一本叫作 *Data Points: Visualization That Means Something*，同样由 Wiley 出版。
- Scott Murray 写过一系列关于 d3 的教程，链接地址为：http://alignedleft.com/tutorials/d3/。由 O'Reilly 出版的 *Interactive Data Visualization* 就是基于这套教程。
- ggplot2 的作者 Hadley Wickham（ggplot2 是 R 中基于 Wilkinson 的作图语法的作图系统）的书 *ggplot2: Elegant Graphics for Data Analysis*（Springer 出版社的 Use R！丛书系列之一）。
- 数据可视化的经典教材还包括 Edward Rufte（统计学家，被认为是数据可视化之父，Mark Hansen 和他是不同年代的人）的 *The Visual Display of Quantitative Information*。该书的重点在于可视化原则和可用工具。第 1 章我们提到的 William Cleveland 有两本关于可视化的著作：*Elements of Graphing Data*（Hobart Press 出版社）和 *Visualizing Data*（Hobart Press 出版）。
- O'Reilly 出版社的一些新书也非常不错，比如 *R Graphics Cookbook*、*Beautiful Data* 和 *Beautiful Visualization*。
- 我们认为数据可视化不能只局限于对工具和统计学的学习。许多艺术院校都会出版关于设计的理论书籍，其他有关于新闻学原则和心理学的内容也与可视化密不可分，了解这些知识对于设计更好的可视化产品都很有好处。
- Jeff Heer 是斯坦福大学的教授，d3 的作者之一（另一位作者是 Micheal Bostock，他之前任职于 Square 公司，后来跳槽去了《纽约时报》）。他十分推介 BretVictor 的讲座"Describing Dynamic Visualizations"（动态可视化简介）。Jeff 说 Bret 为大家展示了数据可视化一个崭新的视角。
- 如果有可能，你要与艺术家或者平面设计师合作。

数据可视化练习作业

选修这门课的学生与正在读这本书的你一样，知识背景差异很大。如果你觉得自己是可视化方面的新手，Rachel 建议你一定要看一看 Nathan Yau 的博客网站，从中选取一两个可视化题目自己动手做一做。在动手的同时，要勤思考，也许你的一些想法会在某些方面改善 Nathan Yau 的可视化结果。

对于课程中其他具备可视化背景的学生，我们建议他们参加 Hubway 的数据可视化竞赛

（http://hubwaydatachallenge.org/）。Hubway 是波士顿市的自行车共享项目。为了更好地宣传这个项目，主办者主动公开了项目的数据集，并组织了这样一场可视化的竞赛。虽然竞赛已经结束，但是数据还是公开的。如此有意思的数据，是练习可视化的大好机会。Rachel 班级里的两位学生，Eurry Kim 与 Kaz Sakamoton 赢得了该项赛事的“最佳叙事性奖”（best data narrarive），Rachel 非常引以为豪。图 9-17 就是他们的竞赛作品，他们利用可视化技术，用一种浪漫温暖的方式讲述了一个关于自行车共享的故事。

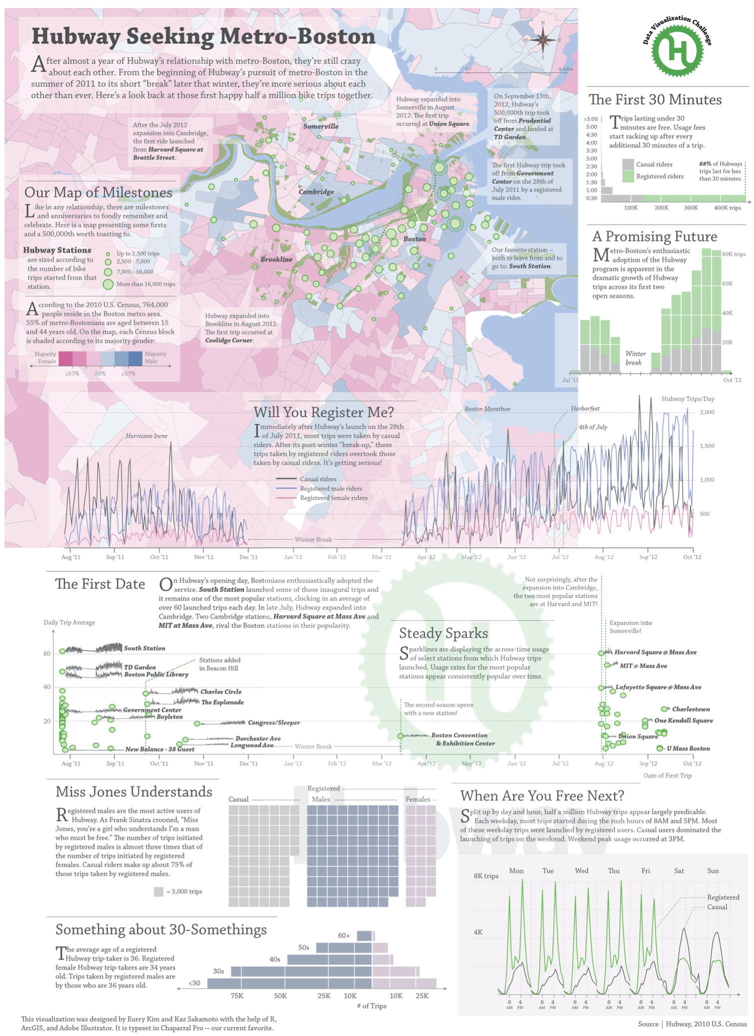


图 9-17：Eurry Kim 和 Kaz Sakamoto 参加 Hubway 公共自行车项目可视化竞赛的最终作品，以及这个项目在波士顿中心区实施的情况（另见彩插图 9-17）

第 10 章

社交网络与数据新闻学

本章要讨论的是过去 5 到 10 年间学术界和工业界讨论的两大热门主题：社交网络分析和数据新闻学。社交网络本身不是一个新的概念（这里的社交网络并不只局限于在线社交网络），对它的研究早在 10 年前就已经风生水起了。它既是社会学的一大研究对象，也吸引了计算科学家、数学家和统计学家的眼球。统计学中的图论就与社交网络的研究密不可分。随着在线类社交网络服务的迅猛发展， 如 Facebook、LinkedIn、Twitter 和 Google+ 等，社交网络研究的可用数据如潮水般涌来，为社会科学的定量研究提供了全新的研究材料。

Morningside Analytics 是一家专注于社交网络数据研究的公司，我们首先会听一听来自他们的声音，并且简要回顾一下有关社交网络的基本理论知识。社交网络数据中蕴含着丰富的故事资源，这是另外一种形式的新闻：数据新闻。我们之前说过，数据科学家的背景理论知识构成就像人的基因组一样，有很多成分：包括数学、统计学、可视化、编程等。然而从事社交网络与数据新闻学研究的科学家所需要具备的知识构成与传统的数据科学家有着细微的差别。虽然两者都要求具备发现问题，用数据解决问题并与人交流发现成果的能力，但数据新闻学家却具备一些独特的研究视角。来自 O’Reilly 的编辑 Jon Bruner 会与我们分享他有关数据新闻学的一些想法和理念。

10.1 Morning Analytics与社交网络

此篇的贡献者是来自于 Morning Analytics 的科学家 John Kelly，他将与我们分享有关社交网络分析的心得。

Kelly1990 年本科毕业于哥伦比亚大学，之后在哥伦比亚大学的新闻学院相继获得了硕士

和博士学位。他的研究领域是社交网络分析与政治统计学。他还抽空在斯坦福大学选修了问卷设计、博弈论等课程。他的硕士论文是与 Marc Smith 在微软一起完成的（http://videolectures.net/marc_smith/）。论文的题目是有关社交网络与政治讨论的联动关系的研究。在上大学之前，Kelly 主要从事音效设计方面的工作。他曾经在哥伦比亚大学艺术学院数字媒体部门做过三年的主管。当年他在越南与妻子度假的时候自学了 Perl 和 Python，然后就成为了一名程序员。

Kelly 觉得，若想在这个领域做好自己感兴趣的事情，就必须具备一些数学、统计学和计算科学方面的知识。这就好比大厨一样，要想做好一桌好菜，没有锅碗瓢盆再好的厨艺也无处施展。

Kelly 感兴趣的是人们为什么会在社交网络上聚集起来？他们在社交网络上都在做些什么事情，这些事情对于政治和公共政策的可能影响是什么，等等。他所任职的公司 Morning Analytics 有很多来自政治组织智库的客户。这些客户找到 Morning Analytics 的目的很简单：了解有关社交网络与媒体活动的趋势及其对政治活动的可能影响。

面对这样的客户，良好的交流与展示技巧是十分关键的。可视化在这其中往往起着至关重要的作用。John 不仅需要具备扎实的理论功底和分析能力，还需要熟练地掌握可视化工具，将分析结果和结论用直观的形式展现给客户。因为客户付钱的原因，并不在于你的分析发现了什么问题，而是你的发现对他们的决策有没有支撑作用。

案例–属性数据与社交网络数据

传统模型所针对的数据通常叫作案例 – 属性数据，每一条案例都对应某个人或者某个事件，相应的人或者事件又由很多属性构成。这样的模型早在 19 世纪 30 年代就被广泛地应用于市场调查研究中了，并且很快蔓延到市场研究的其他领域，包括政治研究。

Kelly 指出，案例 – 属性这样的数据形式给分析带来了巨大的偏差和倾向性。即便这样的数据形式非常适用于现有的数据库系统，收集起来也相对容易，但却极大地束缚了数据分析和研究的广度和深度。

Kelly 提到了另外两位来自欧洲的社交网络分析领域的开拓者，Paul Lazarsfeld 和 Elihu Katz。他们认为社交网络分析不仅要考虑到个人，更要注重人与人之间关系的研究，这是社交网络的本质特征。

社交网络数据分析与案例 – 属性数据分析没有绝对的孰优孰劣，然而有时候应用前者可能会带来更好的效果。比如说，联邦政府想知道人们对于出兵阿富汗的看法，通常的办法是花很多钱做大规模的民意调查，得到的数据是典型的案例 – 属性数据。然而，Kelly 指出，民意不仅仅是单个个体意见的线性组合，要找到整个社交网络中有影响力的组织或者个人，他们的意见往往左右着群众的意见。这样的分析只能通过社交网络分析来实现，传统

的案例 – 属性数据分析则会束手无力。

试想一下，如果现在让你回到 1750 年的欧洲，调查民意并判断未来的政治走向，你会怎么做呢？如果你学过社交网络分析的话，你应该直接分析上层社会的联姻关系，而不是跑到大街上发问卷。

其实在很多情况下，人们习惯于使用案例 – 属性数据是因为这样的数据比较容易收集、储存和分析；即便对于很多问题可能根本就不适用。

Kelly 想要告诉大家的是，现实世界是一张复杂的网络而不是一群独立的个体堆砌而成的。社交网络分析在很多问题上都要优于传统的案例 – 属性数据分析。

10.2 社交网络分析

图论和社会关系计量学对社交网络分析的形成和发展起到了推动作用。欧拉用图论的方法巧妙地解决了 Konigsberg 的七桥问题，而社会关系计量学的出现则得益于 20 世纪 70 年代计算机技术的迅速发展，大型数据的处理从不可能变为现实，其创始人为 Jacob Moreno。

哥伦比亚大学的退休教授 Harrison White 和社会学家 Robert Merton 是社交网络分析的鼻祖。他们认为，对人类行为的分析不能只局限于研究个体的属性，人们之间互相影响所形成的网络（或系统）也应该成为研究的主要对象。

然而到底如何研究这个网络似乎并不是一件很容易的事。Kelly 指出，网络的分析无非是对微观个体和宏观整体的整合性的研究，也就是说既要研究局部个体也要照顾大局整体。

在实际生活中，个体与总体总是通过各种途径联系起来。比如，人们的购买行为形成一张消费网，而人们的竞选和投票行为则构成了一张政治网络。我们亟需合适的工具解析这一张张复杂的网络，这也就是社交网络分析的终极使命。

10.3 关于社交网络分析的相关术语

网络中的基本单位称作*参与者*（actor）或者*节点*（node），可以用来表示人、网站或者其他你能想到的一切事物。这些节点通常在网络图中表示为一个个实心点。节点之间的相互连接的关系称作*关系连接*（relational tie）或者*边*（edge）。比如说在网络图中，如果你对某人点了赞或者和某人互粉，你们之间的关系就可以用一条边连接起来。被边连接的两个个体称作“二分体”（dyad），如果三个节点相互连接则称作“三分体”（triad）。比如说，节点 A 与节点 B 之间有边，节点 B 和节点 C 也有边，则 ABC 三个节点形成“三分体”的必要条件是 A 和 C 之间也有边。

一张网络如果太大，则有必要着眼研究其中的*子群体*（subgroup），也叫作*子网络*，这包括

子群体中的所有节点以及它们之间的边。子网络与母网络从形式上来看并没有本质区别，只是规模大小有别。

如果两个节点之间有*关系*，则从表现形式上来看，它们之间必有线相连。从网络连接的角度来说，对某人点赞与实际生活中与某人住在一起没有形式上的差别，都可以用一条边连接起来。一个*社交网络*就是所有节点与边的集合。

一个社交网络可以很简单也可以很复杂。最简单的网络类似于 Facebook 上的好友网，两个人之间要么是好友要么不是，任何两个人之间都可以成为好友。

稍微复杂一点的是*二分图*（bipartite graph），节点只存在于两个不相交的子集当中，同一个子集的节点之间无边相连。举例来说，一个子集可以是人，而另外一个子集可以是公司，二分图描述的既不是人与人之间的关系，也不是公司与公司之间的关系，而是人与公司之间的关系。如果某个人在某家公司的董事会任职，则他们之间可以用边连接起来。这样的例子举不胜数，比如一个子集仍然是人，而另外一个子集可能是人们感兴趣的社会活动。那么二分图所描述的关系，既不是人与人之间的关系，也不是社会活动之间的联系，而是人与社会活动之间的对应关系。如果某个人对某项社会活动感兴趣，他们之间则存在一条边。

最后一种网络叫作“自我中心网络”（ego network），顾名思义，就是围绕一个节点有很多连接线的网络。以 Facebook 为例，一个人的好友圈就是以这个人为中心的自我中心网络。研究表明，一个人自我中心网络的规模和复杂程度与他的社会经济地位有着直接的联系。通过观察他的自我中心网络就能大致了解他所拥有的社会地位。

10.3.1　如何衡量向心性

给定一个社交网络，人们会问的第一个问题往往是：*谁在这个网络中的重要性最大？*

要回答这个问题，首先需要定义何为重要。重要性在社交网络中又叫“向心性”，这里我们简单介绍几个常用的向心性测度指标，并给出相应的例子。

第一个测度叫作*自由度*，这通常指的是在网络中有多少人与你有边相连。以 Facebook 为例，自由度就是你的好友个数。

第二个测度叫作“紧密度”。如果你的好友与你的连接越紧密，你们的“紧密度”就越高。

为了更好地定义向心性，我们需要定义*连通图*中两个节点间的“距离”。尤其是对于两个非直接相连的节点，距离的定义非常重要。假设节点 x 和 y 之间的距离用 $\mathrm{d}(x, y)$ 表示，最常见的距离可以定义为节点 x 和 y 之间的最短路径的长度。于是，节点 x 本身的“紧密度”可以定义为：

$$C(x) = \sum 2^{-\mathrm{d}(x,y)}$$

接下来介绍另外一种向心性的测度指标，叫作“中间性”（betweenness），它衡量的是在你的网络中，你的好友之间的紧密程度。也就是说，他们之间的最短路径是否通过你的节点，如果通过，说明你在连接这两位好友过程中起到了直接的作用。如果你的中间性指标很高，则代表你对好友的影响很大，起到了信息中转站的作用。

为了精确地定义中间性的概念，我们假设节点 x 和节点 y 之间的最短路径数为 $\sigma_{x,y}$，最短路径中通过节点 v 的路径数为 $\sigma_{x,y}(v)$，那么节点 v 的“中间性”定义为：

$$B(v) = \sum \frac{\sigma_{x,y}(v)}{\sigma_{x,y}}$$

上式的和取自所有与 v 不同的节点（x, y）组合。最后一种向心性测度指标叫作“特征值向心度”，我们会在 10.6.1 节详细介绍。直观的解释就是，如果你与更多的名人们有联系，那么你的特征值向心度要高于常人。谷歌的 PageRank 理论就是这种向心性测度的一个很好的应用。

10.3.2　使用哪种向心性测度

介绍了这么多的测度指标，对于到底该使用哪个指标则是一个头疼的问题。总有一些自称是“权威人士”或者“专家”的人说应该使用某某指标，然而事实上根本没有统一的标准。不同的问题，不同的网络类型，应该使用的指标都不尽相同。

举个例子来说，设想我们要在穆斯林兄弟会成员的博客中找到一个有影响力的博客。一个可能的做法是，先列出网络上最有影响力的前 100 个博客，以自上而下的顺序找到其中与穆斯林兄弟会有关的博客。这样的做法其实完全跟我们想要解决的问题没有关系。你可能找到的是一个与穆斯林兄弟会毫无关系的人，他只是偶尔写了一篇与穆斯林兄弟会有关的博客而已。这个例子告诉我们，如果我们关注的对象只是一小群人，那么我们自然要把视野转到这一小群人所形成的子网络中，而不是在茫茫大网中寻找。

另外需要提示大家的是，根据问题的不同，测度指标的选取也会有所差异。比如说在找博客的时候我们使用的测度指标，如果放在 Twitter 的数据上就可能完全不适用。

一个值得注意的问题是，一些不良用户如果知道了系统使用的是何种测度指标，他们会想方设法提高自己的知名度。做法很简单，如果 Twitter 的系统使用的是特征值向心度，那么不良用户可能会注册多达 5000 个账户，每个账户之间都互相关注，然后再策略性的关注某一个特定账户，这个特定账户的知名度就可能会大大增加。这样的情况肯定是 Twitter 所不愿意并想极力避免的。

如果你用 Python，那么 NeworkX（http://networkx.github.io/）或者 igraph（http://igraph.sourceforge.net/）可以用来计算这些测度指标，在 R 中可以使用 statnet 包（http://statnet.org/）。如果你喜欢用 Excel，可以选择 NodeXL（http://research.microsoft.com/en-us/projects/

nodexl/)。另外我们注意到，斯坦福大学的 Jure Leskovev 正在研发一个基于 C 语言的网络分析包，感兴趣的读者可以关注一下（http://stanford.io/18Pejdt）。

10.4　思维实验

假设你是来自华盛顿某智库的研究人员，手头有 1000 万美元的预算。你的任务是预测埃及这个国家未来的政治走向：包括哪个政党可能执政，埃及在 5 年、10 年以及 20 年之后会变成一个什么样的国家。你所拥有的数据十分广泛，包括埃及公民的教育信息、通话和短信记录、家庭住址、所有政治组织和公司的网络使用情况，以及所有公民的 Facebook 和 Twitter 账户信息等。

在你觉得如何使用这些数据之前，你还应该注意这些信息并不是一成不变的。随着时间的推移，有人会注销 Facebook，政治组织可能会转为地下党（意味着他们的信息会很难获取）等。即便大多数都有 Facebook 账户，但还是有少部分人从来不使用它，这一少部分很可能是你最感兴趣的那部分人。从这个角度来说，通话和短信记录数据可能更加有用。

你可能觉得我们假设的情况很不现实，野心太大。其实不然，德国当年就通过出口西门子制造的宽带设备给伊朗，成功地掌控了伊朗整个国家的宽带使用数据。这样的例子在国家与国家之间经常发生，比如美国就曾经帮助过巴基斯坦，俄罗斯也曾经帮助过叙利亚完成过类似的“间谍”工作。

对于这样一个问题，我们需要改变我们思考问题的方式。面对数据，人们习惯会问：我能从这个数据中得到什么结论呢？这样的想法会严重束缚我们分析和思考问题的广度和深度。这种从数据出发的态度是不可取的。相反，我们应该问类似这样的问题：预测一个国家的政治走向到底要预测哪些具体的指标？我们需要哪些数据支撑我们的分析？这些问题不是以数据为绝对导向的，而是更有深度，也更能带给分析人员有意思的结果。

总之，我们应该以问题为导向，而不是被数据牵着鼻子走。先把问题设定好，再找数据并从数据中找到信息。

10.5　Morningside Analytics

Kelly 在课堂上展示了一张世界上最大的 14 个博客圈的网络地图。要理解这些网络地图，可以想象有一股类似于风的力量将节点（代表博客）往图的边缘推动，而恰好有一股反作用力将这些节点以某种固定形态拉在了一起。这股反作用力就是博客之间的连接关系。图 10-1 表示的就是其中的阿拉伯博客圈。

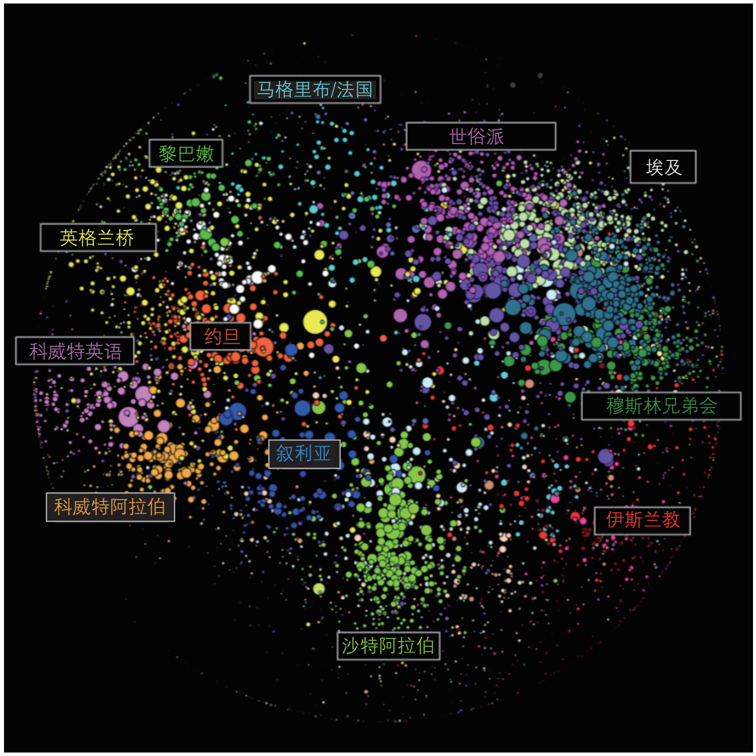


图 10-1：阿拉伯博客圈（另见彩插图 10-1）

图中每个点都代表一个博客，不用的颜色代表不同的国家。点的大小取决于相应博客的向心度的大小，越大则代表着点的直径越大。向心度大代表该博客的连接线越长。这样的网状结构图有助于我们深入了解和洞悉博客圈的情况。

如果只分析每个博客圈中的文章，最典型的方法是用自然语言处理技术（NLP）分析其中的文本，那或许我们会失去很多更有价值的分析对象：博客和博客圈的相互关系。举例来说，应用社交网络分析技术，我们可能会发现不同博客圈的不同关注点，这是纯文本分析技术的盲区。

这样的博客圈可以想象成是某种高维复杂关系在二维平面上的投影。它们到底有什么不同要取决于它们原本内部的复杂关系。从形式上来看，我们似乎可以随便地添加颜色、设定圆圈的大小等。文本分析技术可以在这里帮助我们确认我们分析的结果是否是有意义的，我们所要做的就是如何更好地、定性地解释这些网络。

比如说，法国的博客圈可能讨论的主题是美食，而德国的博客圈主题可能与政治和一些奇奇怪怪的嗜好密不可分，而英国呢？ Cathy 插嘴说道：英国的两大博客主题应该是成人片

和同性恋成人片。应用网络分析，我们可以将这些国家的博客圈主题联系到这些国家的保守性与自由度。

由于俄罗斯严格的网络审查制度，他们的博客圈看起来与其他国家的有着显著的区别。

这里使用的分类算法是 Fruchterman-Reigngold 算法，该算法可以将一些有共同影响的博客圈连在一起，其分类的结果具有优良的可解释性。图 10-2 就是英语圈博客的分类图。

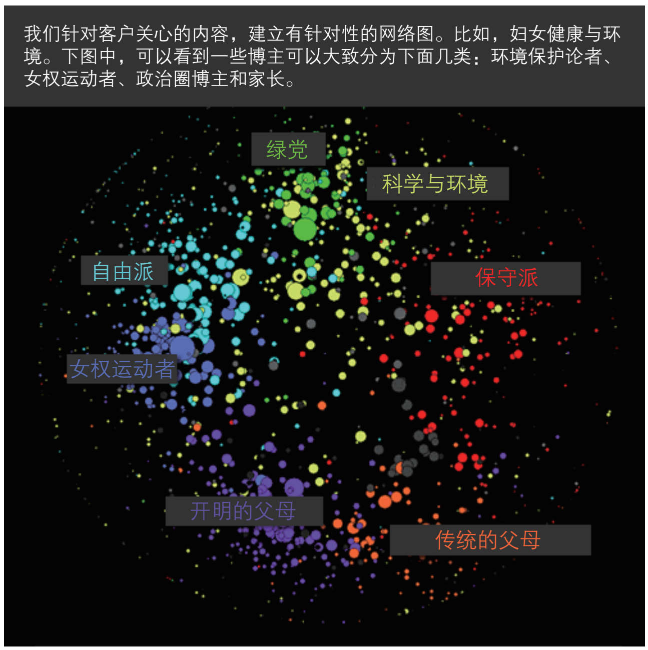


图 10-2：英语圈博客分类图（另见彩插图 10-2）

可视化与中观视角

社会媒体公司要么有数据要么有技术。这些技术基本是一些申请了专利的情感分析引擎或者是一些其他的高精尖技术。也就是说，我们完全不知道这些技术背后的细节，因此要想真正理解你要分析的对象，在使用他们提供的技术的同时，你还需要知道如何使用可视化技术把分析的结果具体化、形象化，以转变成直观的解释。

例子：如果你想要分析大选的结果，你更想知道广大人民群众的看法。因此可以搜索有关“妈妈”或者“体育迷”的博客，因为这些人是老百姓的可能性更大。如果你搜索一些党员的博客，那么他们会写什么内容不看都可以猜出个八九不离十。

Kelly 举了另外一个关于大选的例子。比如说马丁· 路德金的“我有一个梦想”的演讲视频和罗姆尼的竞选视频，在大选期间被博客圈分享的模式是显然不同的。前者会出现在博客

圈的各个角落，而后者只会集中出现在一些支持共和党的保守博客圈内。

但是，如果从视频被分享的次数的直方图来看，似乎后者的分享次数会更多。因此如果不用网络分析的方法而只关注统计量的话，我们很难发现后者其实是人为操作的结果。

Kelly 也为哈佛大学的 Berkman 互联网与社会研究中心工作。她之前在 2008 年和 2011 年分析过伊朗的博客圈。分析结果显示，2008 年和 2011 年的博客圈特征没有显著区别：年轻的反政府民主主义者、诗人（是伊朗社会的重要组成部分）和政府保守派是博客圈的主导群体。

然而，分析的结果同样显示，在 2008 年到 2011 年的三年间，只有 15% 的博客圈没有改变过。

这个例子告诉我们，即便个体的改变是巨大的，但却对整体没有显著影响。也就是说，像社交网络分析这样注重群体网络研究的方式，会比案例 – 属性数据那样注重个体研究的方式带给研究人员更深层次的洞察力。

对于国家和社会这样宏观的研究对象来说，社交网络分析是更加适用的方法：它带给我们是一种介于宏观与微观的独特视角，我们称为中观视角。

10.6　从统计学的角度看社交网络分析

从统计学的角度来看可以将网络看作一个随机数或者随机变量，是由某个随机过程或者某个概率分布形成的。因此，网络也可以生成样本。如果现实生活中的网络是某个随机过程或者概率分布的样本实现，那么我们会问一些统计学中的典型问题。比如说，Twitter 的网络有哪些典型特征？这个网络能很好地代表总体的特征吗？

社交网络分析中的很多问题都来源于数学、统计学、计算机科学、物理学和社会学等学科。它的应用则不仅仅局限于这些学科，甚至可以运用到 fMRI 图像研究、流行病学、社交网络研究（如 Facebook 和 Google+）研究等。

10.6.1　网络的表示方法与特征值向心度

网络节点间的连接方式有两种：有向连接和无向连接。例如在 Twitter 上，如果我关注了你而你没有关注我，这就是一个单向的连接，它是有向的。某些网络是*无向的*，从 Twitter 的角度来说，就是要么我们互相认识，要么我们完全不认识。

一个包含 N 个节点的无向网络可以用一个 $N * N$ 的二元矩阵表示，矩阵中只有 0 和 1 两个值。如果第（i, j）个元素是 1 则代表从节点 i 到节点 j 是相连的。这样的矩阵也叫作*相邻矩阵*（adjacency matrix）或者*关联矩阵*（incidence matrix）。这样的矩阵也可以用来定义有向网络，而无向网络的特别之处在于这样的矩阵是对称的。

另外一种表示网络的方法是使用列表和多元列表。比如说，对于一个节点 i，可以用一个列表列出连接 i 的所有的边，这样的列表也叫作关联列表。这样做的好处是，节点可以有多重属性，而且这样的表示方式可以节省很多存储空间。如果节点有多重属性，那么相应的表示方法叫作多重列表。比如说节点代表人，那么可能的属性包括人的性别、年龄、身高等。如果节点是人的某种行为、习惯或者爱好，那么相应的属性也会随着改变。

连接边也可以有自己的属性，比如可以被赋予权重代表该连接的强度。这种情况下，网络同样可以表示为一个 $N * N$ 的矩阵，只不过应该把 0 和 1 换成相应边的实际权重值。

如果用相邻矩阵表示一个网络，那么就可以定义特征值向心度（10.3.1 节）了。假设相邻矩阵为 $\boldsymbol{A}$，该矩阵的特征值为 λ，对应的特征向量为 $\boldsymbol{x}$，由线性代数理论我们知道：

$$\boldsymbol{Ax} = \lambda \boldsymbol{x}$$

其中：

$$\lambda_i > 0, \quad i=1 \cdots N$$

通过解 $det(\boldsymbol{A} - t\boldsymbol{I})$[1] 便可以求出解特征值和特征向量。通常的做法是将特值从大到小排列，并取其中最大的特征值及其对应的特征向量。特征值代表的就是向心度的大小，而对应的特征向量则是向心度在各个连接上的得分值。某个特征值 λ 对应的特征向量 $\boldsymbol{x}$ 的求法是解下面的齐次方程式：

$$(\boldsymbol{A} - \lambda \boldsymbol{I})\boldsymbol{x} = 0$$

这里得到的特征向量 $\boldsymbol{x}$ 就是我们想要的特征向量向心度。

到目前为止，这些公式只是告诉我们怎么计算得到特征向量向心度，却没有告诉我们为什么是这样。上面的叙述完全是线性代数求解特征值的理论，至于为什么特征向量可以用作向心度的测度指标以及相关的证明和例子，可以参考相关特征值和特征向量的资料。感兴趣的读者可以参考这篇文章：http://goo.gl/UVkLoF。

如果你不喜欢从线性代数的角度计算特征值，也可以用下面一种迭代的方法得到具有最大特征值的特征向量。在迭代之前，假设一个长度为 N 的向量，其元素是每个节点的自由度，通常这些自由度已经被标准化，因此整个向量的元素之和为 1。这个初始向量所包含的信息与节点之间的连接程度没有任何关系。为了得到向心度，对于某一个节点，在下一步迭代的时候将其近邻节点的自由度加总在该节点上，以此类推给所有其他的节点。在一次迭代之后还要进行一次标准化操作以保证整个向量的元素之和始终为 1。反复迭代，每一次迭代的近邻规模都增加一个节点，这样最后得到的向量就是近似为特征值向心度的向量。上面那篇文章也给出了该方法的理论推导。

注 1：这也称作特征多项式。

10.6.2 随机网络的第一个例子：Erdos-Renyi模型

之前已经说过，我们可以将网络看作由某个随机过程产生的样本。具体来说，可以认为网络的连接边的分布是来自于某个分布函数，并假设边之间是相互独立的。

因此，如果有 N 个节点，则一个有 $D = \binom{N}{2}$ 种可能的节点组合（也叫作二分体），每个节点组合之间都有可能有一条连接，或者没有。因此一共有 2^D 种可能的网络。对于每一条连接边，最简单的情形是假设每条边的存在都服从一个参数为 p 的伯努利分布，由此得到的网络模型也叫作 Erdos-Renyi 模型。

伯努利网络

在伯努利假设下，观测到一个所有节点都相互连接的网络的概率为 p^D，而所有节点都不相互连接的概率为 $(1 - p)^D$。这两种网络代表了两种极端，现实中的网络基本都介于两者之间。Erdos-Renyi 模型与**伯努利网络**是同义词。从数学上来看，这样的模型只具有理论研究意义，通常被用作证明更加复杂模型的某种性质。

10.6.3 随机网络的第二个例子：指数随机网络图模型

由于假设条件太不现实，因此伯努利模型很难在现实生活找到可应用的例子。比如说，最常见的好友网络，或者学术界学者之间的合作网都具有明显的传递性（transitivity，也就是说，如果 A 认识 B，B 认识 C，那么 A 也认识 C）、聚类性（clustering，有相同特征或者兴趣的人倾向于一类，从网络形式上来看，就是在母网内有很多抱团的小网络）、相互性（reciprocity 或者 mutuality，也就是说如果 A 加了 B 为好友，那么 B 也会加 A 为好友）、中间性（betweenness，通常是有影响力的节点才具备此特征，它在网络的信息流动中扮演着重要中间人的角色）。所有这样特性的存在都说明我们需要更加复杂的模型。

类似于上面的网络特性都可以用数学语言表示。比如说，传递性就可以表示为网络中的三角形的个数。

指数随机网络图模型（ERGM）是社会学中广泛使用的网络模型，它可以涵盖大多数我们讨论过的网络特性。

ERGM 的研究对象是网络中的某些典型变量：比如网络中三角形的个数、连接边的个数、双星的个数（双星指的是某个节点有两个连接边，因此如果一个节点的自由度为 3，则它包含 3 个双星）。这些变量用 z_i 表示，并且假设其分布的参数为 θ_i。例如，用 z_1 表示网络中三角形的个数，并且该变量的参数为 θ_1。如果 θ_1 是一个较大的正数，则代表该网络中有较多的三角形，其节点之间具有较强的传递性。

还有一些较为复杂的变量，比如 k 星（k-star，与双星类似，k 星表示某个节点具有 k 个连

接边，一个自由度为 $k+1$ 的节点有 $k+1$ 个 k 星结构）。一个较为复杂的 ERGM 模型可以表示为：

$$Pr(Y=y)=(\frac{1}{k})(\theta_1 z_1(y)+\theta_2 z_2(y)+\theta_3 z_3(y))$$

该模型的含义是：对于一个网络 Y，出现我们观测到的网络 y 的概率可以表示一系列网络变量的线性组合形式。

从形式上来看，伯努利网络是 ERGM 的特殊形式，也是最简单的形式。伯努利网络的变量就是网络中边数。

ERGM的推断问题

从统计学的角度来说，理想化但非常不现实的情况是，对于某个网络 Y，我们可以观测到一系列的样本网络，$Y_1,\cdots,Y_n$，其中 n 是样本量。每一个样本网络都假设有 N 个节点，并可以用一个相邻矩阵表示。

给定这些样本网络，我们假设它们是相互独立的，并来自于同一个概率分布模型。在这样的假设下，ERGM 的推断是可能的。[2]

以伯努利网络为例，假设某条连接边存在的概率为 p，那么观测到某一个特定样本网络集合的似然概率可以表示为：

$$L=\prod_i^n p^{d_i}(1-p)^{D-di}$$

其中 d_i 是第 i 个样本网络中观测到的连接边的个数，而 D 是网络中所有二分体的个数。由此，p 的估计值为：

$$\hat{p}=\frac{\sum_{i=1}^{n} d_i}{nD}$$

然后，从实际情况来看，我们不可能观测到一系列的样本网络，通常只有一个样本，也就是说有效样本量为 1。我们用 1 个样本估计了模型中的参数，这看起来似乎不可思议。对于伯努利网络模型来说，p 的估计值就是样本网络中连接边的个数与二分体总是的比例值。这看起来很合理。

但对于更加复杂的 ERGM 模型来说，一个样本对于参数估计来说是远远不够的。当然，我们可以使用类似伪似然估计（pseudo-likelihood estimation procedure）这样的估计方法。但是即便是这样，还是会有很多困难。Mark Handcock 2003 年的论文“Assessing Degeneracy of Statistical Models of Social Networks”（“论社交网络统计模型的退化性”，参见 http://

注 2：也就是说，其中的参数是可估计的。

citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.81.5086）详细地讨论了该问题。他指出，即便是使用基于模拟的方法（比如MCMC，马尔科夫链蒙特卡洛方法），也很难避免"推断退化"问题（inferential degeneracy）。也就是说，所估计出来的网络很可能是一个退化的网络（完全相连或者完全没有连接的网络），或者说估计的结果没有一致性等很多问题。

关于随机图模型的其他例子：隐空间模型与小世界模型

由于指数随机网络模型中可能出现的模型退化以及模型估计不稳定的问题，研究人员提出了一种新的图模型：隐空间模型（latent space model）。Peter Hoff的文章"Latent Space Approaches to Social Network Analysis"（"基于隐空间模型的社交网络分析"，参见http://1.usa.gov/GzAT1Z）是该研究的开山之作。

隐空间模型主要解决一个问题：网络中有很多因素是不可观测的，这样的不可观测因素既是客观存在的，也是非常重要的 。传统的模型不考虑它们的存在，隐空间模型可以很好地捕捉到这些因素。比如说，Facebook上的好友网络，我们观测到的是人们之间虚拟的好友关系，但是至于这些人住哪、他们到底为什么会成为好友，单单从网络上来看是无从知晓的。

Watts和Strogatz于1998年提出了小世界模型（参见http://en.wikipedia.org/wiki/Watts_and_Strogatz_model），该模型要估计的网络介于完全随机网络与完全不随机网络之间。它想在现实网络世界中验证六维自由度理论。然而，人们对此模型的批评主要集中于它的过度均匀假设。因为现实世界的网络是没有尺度特性的，并且非均匀的。

除了上述模型之外，还有很多其他的图模型：比如，马尔科夫随机场模型（Markov random field）、随机块模型（stochastic block model）、混合会员模型（mixed membership model）、随机块混合会员模型（stochastic block mixed membership model）等。关于随机块混合会员模型，可参考Edoardo Airoli等人的论文"Mixed Membership Stochastic Block Models"（"随机块混合会员模型"，参见http://dl.acm.org/citation.cfm?id=1442798）。

下面列出一些关于社交网络分析的书籍，感兴趣的同学有时间可以读一读：

- *Networks, Crowds, and Markets*（《网络、人群与市场》，原英文版由剑桥大学出版社出版），作者是来自于康奈尔大学计算科学学院的David Easley与Jon Kleinberg；
- *Mining Massive Datasets*（《大数据挖掘》，原英文版由剑桥出版社出版）一书中关于社交网络图模型分析的章节，作者是来自斯坦福大学计算科学学院的Anand Rajaraman、Jeff Ullman和Jure Leskovec；
- *Statictical Analysis of Network Data*（《网络数据的统计分析》，原英文版由Springer出版社出版），作者是来自波士顿大学的Eric D. Kolazcyk。

10.7 数据新闻学

来自 O'Reilly 的 Jon Bruner 与我们分享了有关数据新闻学的内容。Jon 之前在福布斯杂志做数据编辑，他的数据科学技能十分全面，他的研究和出版内容也都与数据相关。

10.7.1 关于数据新闻学的历史回顾

数据新闻学已经发展了相当长的时间，比如基于 Excel 的自动报告系统就是某种形式的数据新闻学。但是这样的自动报告系统一直是 Excel 一统天下，即使在今天，如果你能写一手不错的 Excel 程序，也有人愿意花大笔钱雇你。

API 的出现以及计算机价格的平民化改变了这个现状，各种形式的自动化报告工具如雨后春笋般地出现。现在的数据分析工作，即便数据较大，一个人也可以单枪匹马地在个人笔记本上完成。越来越多的人开始具备基本的编程素养，甚至有些作家都可以写一手漂亮的数据分析程序。一些英语专业的学生也会花很多时间研究计算机和编程。更有意思的是，计算机出身的人也可以写出一手很好的文章。从趋势上来看，人们都变得越来越全面了。

在《纽约时报》这样的大型新闻出版机构，数据新闻学被细分为很多微部门：平面设计、交互设计、数据库工程、爬虫工程、软件设计、领域专家和写手，等等。有些人只负责提出问题，而真正动手干的又是另一批人。比如，Charles Duhigg 任职于《纽约时报》，他最近收到了纽约州议会关于信息自由法案（FOIA）的问案，因为他是这个领域的专家，因此他清楚地明白就该方案应该提哪些相关的问题，但是他不会着手于具体的数据分析。分析的活是留给另一帮人干的。

如果是在一个小公司里，情况就大不同了。在《纽约时报》的大楼里，数据新闻部门有差不多 1000 个员工，而在《经济学家》杂志社，他们有 130 人，《福布斯杂志》有接近 80 个人。如果你在一个小公司工作，那么很可能所有的脏活累活你都得一个人全揽下：你要想该问哪些问题、该如何收集数据、自己做数据分析、自己写数据分析报告。当然，如果有两三个同事帮忙最好，因为每项工作都是技术活。

10.7.2 数据新闻报告的写作：来自专家的建议

Jon 毕业于芝加哥大学，主修专业为数学。毕业后加入了《福布斯》杂志社从事写作工作。由于工作的要求，他也慢慢地开始做一些定量分析的工作。在报道亿万富翁或者政治家的社会贡献时，他也时常会用到一些图分析的工具。

在课堂下，Jon 以自己的数据科学背景为例，为大家解释了“数据新闻学”的含义。

首先，数据新闻学需要大量的数据可视化工作，因为可视化是直观地报道和解释数据的最有效的工具。计算科学方面的知识对于精通数据新闻学也十分重要。因为时间就是生命，

数据新闻要求对分析的工具掌握娴熟，能够熟练快速地处理好原始数据。数据新闻工作者要面对形形色色的数据，这就要求他们能够熟练地使用像 Python 这样的工具处理原始数据。Jon 自己就精通 JavaScript、Python、SQL 和 MongoDB。

统计学对数据新闻学的重要性同样不言而喻。统计学是数据分析的基石，是我们思考的方式。比如，某篇报道中可能会写“Twitter 上平均每个女性有 250 个好友”，这里使用的是平均值。如果用中位数，则平均每个女性的好友数为 0。因为原始数据是严重不对称的，使用不同的统计量则可以讲出不同的故事。

Brune 说他本人在机器学习方面是一个不折不扣的新手，但是掌握一些机器学习的理论和技术对于数据新闻学同样重要。这也与你工作单位的规模有关系，如果你任职于政府部门或者大型的日报集团，那么你需要是某个领域的专家。然而，如果你在一个小单位工作，就像我们之前说的，你什么都得懂。

另外两项对于数据新闻工作者来说至关重要的技能是：交流与展示。如何把一个个复杂的故事以通俗、容易理解的方式展示给读者，是数据新闻工作的基本要求。同样，数据新闻工作者要时刻准备着回答读者提出的各种问题，把问题转化成数据分析任务，再将分析的结果以同样通俗和容易理解的方式反馈给读者。

Jon 给大家的最后一条建议是：时刻准备好改变主意！数据新闻工作类似于探索性数据分析，我们要自己做分析也要和领域专家打交道。信息会以一种非常难以预料的方式砸向我们，因此我们应该时刻做好改变主意、改变思考和分析方向的准备。

第 11 章

因果关系研究

到目前为止，本书讨论的模型和一些实例都是针对预测问题的。比方说，第 8 章我们介绍了如何预测人们对某件事物的偏好：比如一部电影或一本书。这样的模型可以纳入成百上千个特征变量，再利用变量选择的方法筛选出对于因变量最为重要的那些特征变量。模型的终极目标是最大化模型的预测准确度。在模型优化过程当中，变量本身的含义和解释就显得无足轻重了。尤其是当模型中的变量个数很多时，我们不可能逐一地解释每个变量的含义。

也就是说，如果建模的目的是最大化模型的预测精度，那么你大可不必花很多心思在变量的解释上。例如，一个亚马逊的图书推荐模型可能包含这样一个变量，即“你是否读过 Wes McKinney 的 O’Reilly 系列书 *Python for Data Analysis*”，这个变量对于预测你是否会读这本书当然有用。但是，是否读过这本书就代表你会买下它？这可说不定。而且这个解释本身听起来就像是一段同义反复。当我们的建模标准时预测准确度时，我们可以这么做，不必担心如何理解或者解释变量间可能存在的因果关系。但是，如果你真的想要构建研究因果关系的模型时，就不能这么干了。

实际生活中不是所有的问题都是预测问题，你可能真的想要研究变量之间的因果关系。到底什么是因果关系？说白了，如果你想要做出某种行为导致了某个结果的论断，这便是因果关系推断。因果关系模型并不是一套完全不同于预测模型的统计方法。恰好相反，它其实是根植于传统预测模型（如逻辑回归、线性回归）的框架内的。但是，你的思路和目标就不再是优化模型以提高预测的准确性了，而是尽力分离出变量之间的因果关系。

这一章我们就要着重探讨因果关系的建模。我们特别邀请了这个领域的两位专家：Ori Stitelman 和 David Madigan。Madigan 也是下一章的主要贡献者（我们会在下一章详细介绍

他）。但是想要想理解下一章的内容需要仔细研读本章。Ori 是 Wells Fargo 的一名数据科学家，他之前在一家法律事务咨询事务所工作，随后在加州大学伯克利分校获得了生物统计学的博士学位。他的主要工作是从数据中攫取信息、找出故事，并与相关领域的专家交流。在与形形色色的数据库打过交道之后，他提出了“数据直觉”这一概念。

11.1 相关性并不代表因果关系

确定两个变量间的因果关系是统计学的一大难题。想一想，当你说一个事件会导致另一事件的发生的时候，需要多大的勇气？实话来说，确定因果关系确实是一项艰巨的任务！

假设我们发现了冰激凌的销售额与泳衣销售额之间存在相关性，图 11-1 画出了它们的时间序列图，图中可以看到它们之间有明显的相关关系。

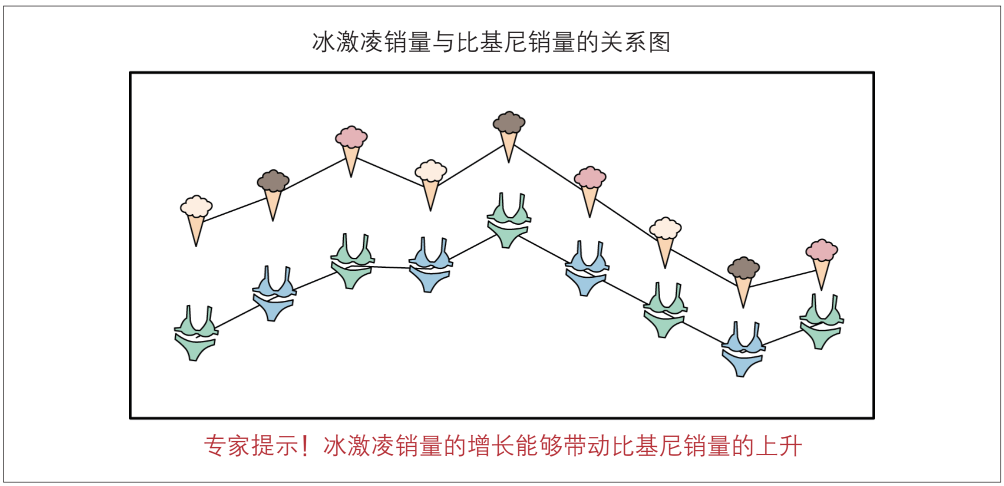


图 11-1：冰淇淋销售额与泳衣销售额的时间序列图

虽然该图显示二者密切相关，但这并没有建立起任何的因果关系。现在重新考虑二者的关系，许多解释都可能符合图中二者的关系：也许是当人们穿着泳衣时，就特别想吃冰激凌？也许是每次吃冰激凌时，人们都会换上泳衣？又或者是我们没考虑到的第三个变量（如高温）导致上述二者同时发生？这些听起来都有可能，尤其是第三个关于高温的推测。因果关系推断就是为了理解在哪些情况下，变量之间的这种相关关系可以确定为因果关系。

11.1.1 对因果关系提问

对因果关系最自然的提问方式是：x 对 y 的影响是什么？

来看几个例子：“广告对消费者行为的影响是什么”“药物对杀死某病毒有效吗”，又或者更一般的说法，“实验对结果的影响是什么”。

“实验”与“非实验”这两个术语来自生物统计学、医学与临床学领域，指的是患者是否接受了某种医学治疗或者处理。关于医学（流行病学）的例子会在下一章详细讨论。本章之后的讨论会经常使用“实验”和“处理”这样的术语。这些术语也经常出现在统计学和社会学研究的文献中。

实话说，因果推断中的参数估计是非常困难的。比如说，广告到底有没有作用？它的作用有多大？这是一个典型的因果推断问题，但是却基本不可能有精确的答案。因为其中因果关系的强度实在是太难估量了。人们通常花大力气研究那些简单易测的变量，但这些变量却并未能测出他们想要的东西，而大家不管三七二十一，都根据这些变量的研究结果做出决策，这样的研究是非常不负责任的。比如，营销人员会因为销售业绩好而受到公司的奖励，因为公司认为他们的营销努力为公司带来了更高的销售额。这是一个典型的因果关系推断，但是其中一个值得怀疑的地方是，销售业绩好可能是因为那些消费者本来就有强烈的购物欲望，跟营销人员的工作没有关系。这里面就有一个“干扰因子”的问题，它是因果关系推断的核心概念，下面我们就用一个更加详细的例子解释这个概念。

11.1.2　干扰因子：一个关于在线约会网站的例子

让我们来看这样一个例子，是有关一个叫 Frank 的寂寞的家伙在网上约会的事。假设 Frank 在一个约会网站上发现了心仪的对象。为了说服她出来跟他约会，他首先要写一封能引起她兴趣的邮件。他该在这封搭讪邮件里说些什么？如果 Frank 看过这位姑娘的头像，觉得她长得很漂亮。那么他能直接在搭讪的邮件里夸赞对方长得漂亮吗？也就是说，在搭讪邮件里就直接夸赞对方漂亮对 Frank 有好处吗？对方会买账吗？

理论上来说，Frank 可以做一个随机实验。假设他心仪的对象有很多，这些对象构成一个样本。随机实验的做法是，将样本随机分成两半，一半的样本 Frank 会在搭讪邮件中称赞说她们很漂亮，而对另一半的人不做任何夸赞。如果前一半的对象反馈要明显好于后一半，那么就可以确认这样的搭讪技巧是有效的。

然而，不管是什么原因，Frank 并没有这么做。这个做法听起来就挺疯狂的。于是得由我们来决定，对 Frank 而言，这样的搭讪风格是否有效。Frank 能否搭讪成功现在完全取决于我们。

让我们先将这个因果问题明确地提出来：Frank 在搭讪邮件中告诉一位姑娘她很漂亮会对他的搭讪成功率产生什么影响？换句话说，这里的“实验”，或者“处理”，是 Frank 通过搭讪邮件告诉一位姑娘她很漂亮；而“结果”是这个姑娘有无积极的回复。这里的“控制实验”就是 Frank 在搭讪邮件中没有提到“漂亮”的事，而是扯了一些别的。

在这个例子中，其实还有很多因素没有考虑。例如，我们并没有提到 Frank 的为人。他也许是个怪胎，很不讨人喜欢。因此无论他说什么都没有姑娘愿意跟他约会。又或许他压根不会写“漂亮”这个词。相反，如果他是个帅哥、暖男或者名人，则不管他说不说，女方可能都愿意与他约会？另外，考虑到大部分的约会网站给男性与女性联系对方提供了同等便利的条件，有些姑娘可能会主动找上 Frank，无论 Frank 有没有事先给她们发邮件。因此，从这些因素来看，因果推断其实是非常复杂的。

11.2 OK Cupid的发现

OK Cupid 是一家在线约会网站，他们利用近 50 万会员的数据，分析了一些常用词和短语在第一次邮件（搭讪邮件）接触的时候对回复率的影响。分析的结果可见图 11-2。

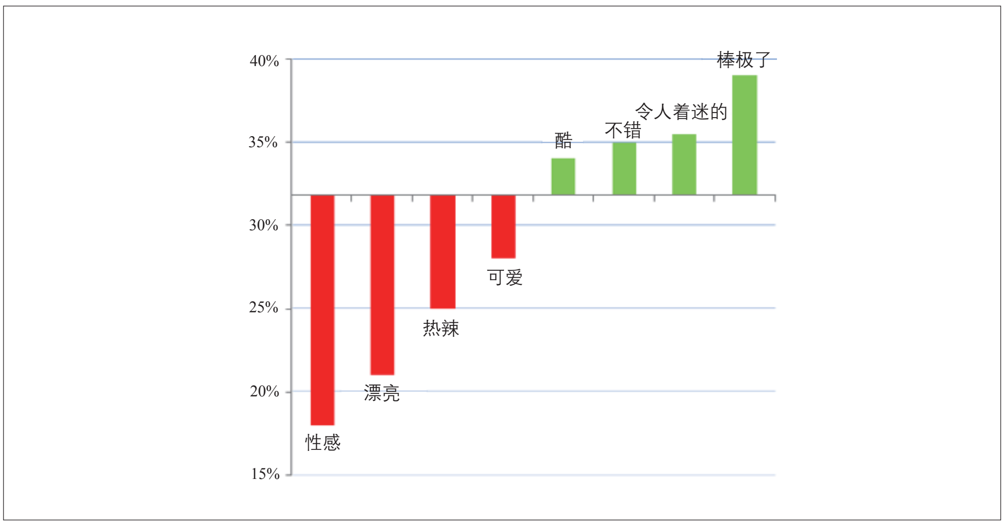


图 11-2：OK Cupid 的研究发现，在第一次接触性对话中使用“漂亮”一词不利于得到积极的答复（另见彩插图 11-2）

Y 轴表示回复率。平均来看，所有邮件的回复率约为 32%。然后，他们将这些邮件按关键词，如“漂亮”或“惊艳”来分类，并观察各类邮件的回复率。如果用条件概率来表达上述结果，可以说他们估计的结果是：P(回复) = 0.32，而 P(回复 | “漂亮”) = 0.22。

上图遗漏了一个重要信息：那就是每个分类样本的的样本量。有多少人在首次接触性对话中使用了这些词？这个问题并不会改变所得到的结果，但它却有助于我们弄清楚图中那条位于 32% 的横线是否是根据每个分类样本的样本量计算的加权平均值。

他们把上述发现总结为搭讪的第一原则："避免过度恭维。"他们还把这条发现发在了公司的博客上，题为"在线约会应该如何搭讪"。文中说到："你也许认为人们都喜欢被'光彩照人''漂亮'及'性感'这样的词语包围，但从在线约会搭讪的数据分析来看倒并非如此。在见面之前，用这些词语搭讪往往会事与愿违。另外，当你告诉一位女士她很漂亮的时候，很可能是你不够帅。"

从统计学的角度来说，上面的例子叫作观察性研究。观察性研究指的是数据的生成过程没有受人为干扰，是自然生成的。这与人工设计的实验正好相反——在实验中各种因素都被人为控制，以研究某一个特定因素对实验结果的影响。从观察性研究的角度来说，能否根据上图就推断，在邮件中使用"惊艳"（fasinating）可以提高回复率，而使用"漂亮"（beautiful）就会降低回复率呢？

在回答这个问题之前，先考虑以下三方面的问题。

首先，在第一次搭讪的时候用"漂亮"这个词，往往在暗示对方自己也很帅？而对方可能会认为此人太过多情。另外，"漂亮"不仅可以用来形容女性，也同样可以用来形容食物、衣物等。因此，也要考虑词语使用的具体语境。

其次，上面的两点在因果推断的时候都要考虑到。但是即便语境本身很重要，但是从 Frank 个人的角度来看，其实关系并不大，因为不管他多么多情，或者他用"漂亮"赞美的是其他食物，这对于 Frank 搭讪的对象来说都是出自 Frank 之手，因此对于 Frank 本人来说，这些因素并没有干扰因果推断本身。

最后，要考虑的最重要的问题是，收到包含"漂亮"一词邮件的人很可能比较特别，她可能因为头像比较好看而经常被人搭讪。因此，这部分女性每天会收到成堆的邮件，而只有精力回复其中很少一部分，因此像 Frank 这样的人收到她们回复的可能性就更小了。

事实上，如果"漂亮"可以被完整地定义，它在这个例子中可被视作一个干扰因子。也就是说，如果这个女士真的"漂亮"，会同时影响到 Frank 是否给她发邮件以及她是否会回复 Frank。当一个变量同时影响到"实验"本身，以及"实验"的结果时，它就是一个干扰因子。

如果考虑干扰因子的影响，OK Cupid 的研究以及他们对上图的解释可能是完全错误的。但由于我们没法得到实际的数据，也不能妄下定论。但是，我们可以讲清楚我们需要什么样的数据，以及如何合理地分析数据。他们的分析和解释也许是对的，但单从一张图很难做出合理的因果推断。

11.3 黄金准则：随机化临床实验

我们到底应该怎么做才能确定变量之间的因果关系呢？

确立因果关系的黄金准则是使用随机化实验。顾名思义，随机化实验的关键在于随机化：样本被随机化为两个子样本，一个作为实验组（接受处理），另一个作为控制组。随机化之后，两组样本的表现差异就可以视作是“处理”因素引起的。从统计学角度来看，随机化保证了两个子样本都是来自同一个总体的同质样本，因此对于两个子样本来说，潜在干扰因子的可能影响是同等的。这从理论上排除了所有潜在干扰因子的影响。

随机实验的效果很好，因为在随机化的过程中，所有可能成为干扰因子的因素都被排除了（比如是否有吸烟史）。随机化保证了有抽烟史的人将会以同样的概率被分到两个子样本中，于是“吸烟史”这样一个干扰因子就被随机化排除了。

随机实验的绝妙之处在于，不单是我们所能想到的，就连那些我们很难考虑到的无数其他干扰因子的影响，也被排除了。

因此，虽然我们可以通过算法针对某些变量找到一些不错的划分，但是这些划分不可能对所有变量都有同样好的效果。这也正是我们需要随机化的原因，因为随机化无论对于我们能考虑到的变量还是没有考虑的变量都有同等的效果。

随机实验在医学研究中也有自己的软肋。根据医学研究的“临床均衡”原则，只有当医学界确实不清楚哪一种治疗方法更好时，随机化分组才是道德上可以接受的。如果研究人员基本确信某药物对某疾病有效，而将一部分人随机化分组到控制组中（也就是说，不给予该药物治疗），这是不符合医疗道德的。

举个例子。为了找出抽烟与心脏病之间的联系，我们不能随机地选出一部分人并建议他们抽烟，因为抽烟有害健康是一个公认的事实。同样，采取随机临床实验研究吸食可卡因与婴儿重量的关系也犯了道德禁忌；研究饮食与死亡率的关系也同样十分棘手，因为这些都直接影响到人们的生命健康安全。

另一个问题是，随机临床实验通常都耗资巨大并且十分烦琐。然而，矛盾的是，如果不做随机临床实验又可能导致错误的研究结论，代价甚至更加昂贵。

当然，有时候不是想做随机实验都可以做的，在很多情况下，随机实验甚至是不能实现的。比如在 OK Cupid 的例子中，我们明明知道存在大量的干扰因子，却无法应用随机实验。为什么呢？想象一下，如果系统随机的给女性会员发送赞美她们的邮件，那么 OK Cupid 可能第二天就倒闭了。

总之，当随机实验的条件满足时，它是解决因果推断问题的黄金法则。然而，随机实验也常常由于道德和现实条件的限制而变得不可行。

平均与个体

随机临床实验衡量的是某一种药物对所有人的平均效用。要想研究关于男性、女性，或者某一年龄段的组别平均效用，则需要采取分组的方法。但即便分组得很仔细，所得到的效用也仍是平均意义上的。换句话说，我们的研究还不能将实验的效用具体到某个人身上。最近一段时间，伴随基因组技术的出现，个性化临床研究开始萌芽并显示出其独特的应用价值。以之前 OK Cupid 的研究为例，研究的结论是所有男性的平均效果，而不能单独应用在 Frank 身上。

11.4 A/B 测试

在软件领域，随机实验也称作 A/B 测试。事实上，我们发现，如果对工程师说“实验”这个词，那意味“尝试新的事物”：如让用户实验同一个产品的不同版本，以推断用户对软件版本的喜爱偏好；而不一定是指一种统计分析方法。A/B 测试其实十分易于理解，操作起来也不是很麻烦。事实是，一个简单的 A/B 实验可以用一个简短的配置文件，外加一个调整参数（如颜色、外观、软件版本等）即可实现。因此，在科技公司内开展 A/B 测试在某些方面比进行一个临床实验要简单得多。并且，由于和人的生命健康关系不大，因此实验的道德和实际风险都很小。在医学随机临床实验中，我们不能选择让一部分人用药而另一部分人不用；而如果实验的平台是整个互联网，我们可以决定给用户展示什么或者不展示什么，这基本不会引起任何监管的问题。但这些也都不是绝对的，即便是科技公司和互联网公司，在 A/B 测试时也需要考虑很多问题。

A/B 测试理论上很简单，但放到公司层面，实施起来却没有那么容易。公司的产品部门会有很多小组，每个小组否负责产品某方面的特性。如果这些小组在 A/B 测试的时候没有协调好，那么 A/B 测试的效果会大打折扣。比如，用户界面小组总是想测试用户对字体字号的喜好，于是会用到 A/B 测试。与此同时，内容评分小组想要改善某个推荐算法，也会用到 A/B 测试，而广告小组则会想方设法提高广告系统的盈利能力，也用到 A/B 测试。在 A/B 测试的时候，各个小组关心的结果是一样的：某项改动能否带来更多的用户点击。在测试的过程中，如果某个小组发现，用户点击率确实在某项改动之后有了显著的增加。然而，这个增加的效果到底应该归功于用户界面小组，还是另外两个小组呢？如果小组之间沟通不畅，协调不够，则 A/B 测试的结果可能会无法溯源。用户界面小组认为是字体的改变带来了更多的用户点击，但实际上很可能是由于推荐算法的改变或者广告系统的升级等多项改变联动引起。

A/B 实验的的基本构建有很多方面是要细心考虑的，Diane Tang 等人于 2010 年所写的论文“Overlapping Experiment Infrastructure: More, Better, Faster Experimentation”（“重复实验构建：更多、更好、更快的实验”）对此有过详细介绍。我们从该论文中摘取一段，与大家分享。

摘自“重复实验构建：更多、更好、更快的实验”

如题，我们进行实验基础构建是为了实现更多、更好、更快的目标。

- 更多

  实验需具备可扩展性以同时运行更多的实验。然而，灵活性也十分重要，因为不同的实验需要不同的参数设置以及不同的样本量以更好的估计统计显著性。某些实验可能只与程序运行的某个子段有关，而其他实验可能与整段程序有关。

- 更好

  应该尽量避免明显无效的实验，低效的实验也应该得到尽早地优化或者直接剔除(比如错误代码或者效果明显很差的代码等)。应该用一套标准的程序检验实验效果的好坏，并比较不同实验效果之间的优劣。

- 更快

  构建实验的方法应该简单易行，以便非工程师不用写代码也能直接上马实验。实验模型的评估应该快准狠。如果是简单的重复性实验，速度是关键。理想的状态下，系统应该不仅能够支持实验，还能够系统的控制项目的预热与爬升。也就是说，实验或者模型的某种变动应该以一种系统性的、容易被理解的方式，逐渐地渗透到项目的其他部分。

一个项目的实验基础架构通常会由一个很大的团队在背后支撑，全天候地做着各种各样的分析工作。虽然是基础架构，却绝非易事。随着社交网络的迅猛发展，实验的基础架构变得越来越复杂。因为网络的先天相关性为实验设计中的独立性假设提出了巨大的挑战。比如说，Facebook 设计了一个实验，Rachel 被分在了“实验组”，实验处理的内容要求 Rachel 发表一些特定内容的博客，而 Cathy 被分在了控制组。实验的随机化原则告诉我们，Rachel 和 Cathy 是相互独立的。但事实上呢？由于社交网络的关联性，Rachel 发的博客，Cathy 即便是在控制组也难免会看到这些博客，因此她们不是完全独立的：Cathy 因为网络的关联性，也接受了一定程度的“处理”。社交网络为很多基础研究都提出了新的课题，有待科学家们进一步研究。

11.5 退一步求其次：关于观察性研究

虽然一般情况下因果关系推断的黄金准则是采用随机实验或 A/B 测试，但正如我们反复强调的，它们并不总是可行的。有时候我们不得不退而求其次，用观察性研究的方法解决问题。

让我们先介绍它的定义：

> 观察性研究是当控制实验（随机实验）不可行时而采用的一项分析因果关系的实证性研究方法。

许多数据科学研究都是围绕观察性数据展开的。前面所讨论过的 A/B 测试是个例外。很多时候，你能使用的数据就是你所观察到的数据。很多时候，我们没有能力让时间倒流，或者让同样一件事情重复发生。例如，对于总统大选这样的事件，我们只有观察性的数据，不可能进行任何实验。

众所周知，实验性的数据要优于观察性的数据，因为在实验中，很多因素可以被人为地控制，观察性的数据则不然。然而很多的实验从道德、成本上来考虑是不现实的。因此我们不得不分析手头能够得到的、观察性的数据。观察性研究就是在这样的一个背景下，为了研究因果关系而退而求其次的研究方法。

即便你可能毫不关心因果推断的问题，而只在意模型的预测效果，预测模型使用的仍然是观察性的数据。观察性研究中可能遇到的问题，在预测模型中同样会遇到。

11.5.1 辛普森悖论

首先，观察性研究中存在着许多陷阱，辛普森悖论就是其中一个。

图 11-3 是一个简单的散点图，你可以找到一条最佳拟合线来描述是否食用高剂量的某种“不良药剂”会提高犯心脏病的概率。

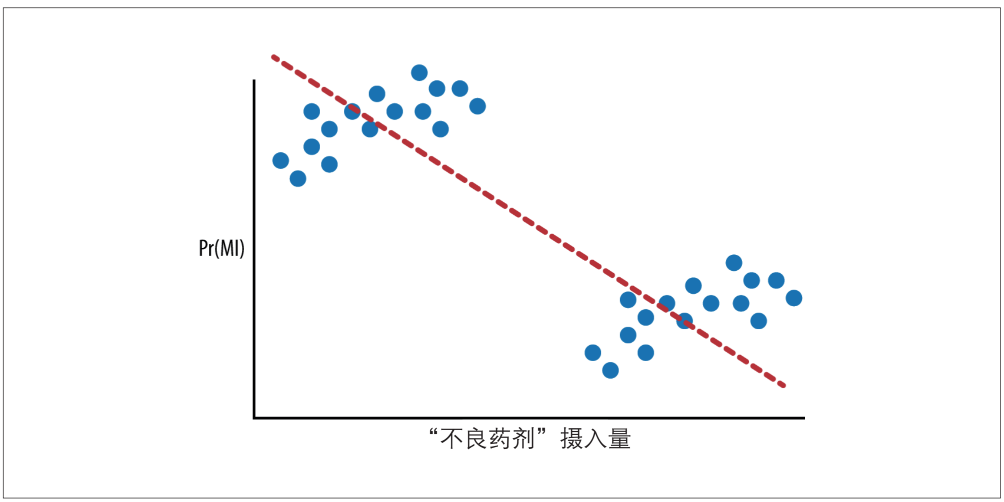


图 11-3：患心脏病（简称为 MI）的概率与某种“不良药剂”摄入量的关系散点图

从最佳线性拟合的角度来看，上图似乎表明食用剂量越高，犯心脏病的可能性越小。但是，图中的数据形成了两个明显的聚类，如果进一步分析这两个聚类，反而会得出截然相反的结论，这从图 11-4 可以看出。

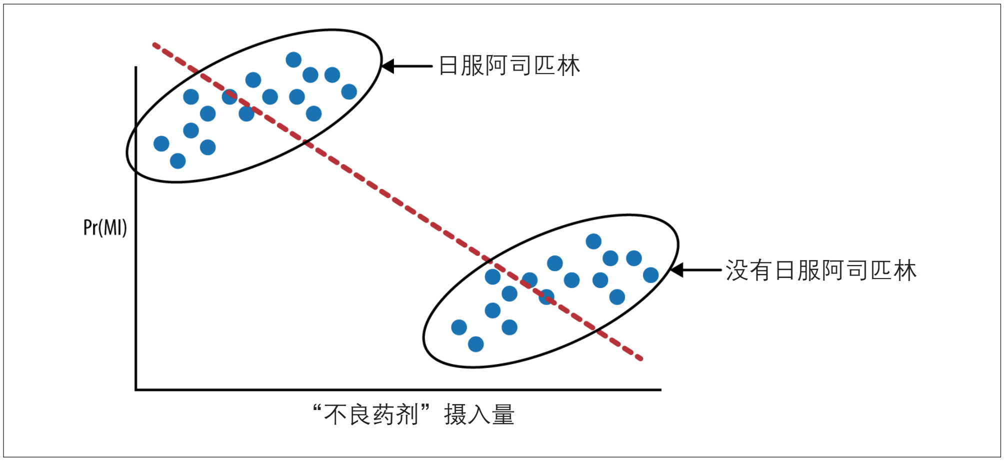


图 11-4：患心脏病（简称为 MI）的概率与某种“不良药剂”摄入量的关系散点图，此处考虑两个聚类，一类是日服了阿司匹林的患者，一类是没有日服阿司匹林的患者

这幅图是有意设计的，因此聚类的问题看起来十分明显。但当数据是多维时，你很难一眼就看出来其中可能存在类似的问题。

在这个例子中，我们可以说“日服阿斯匹林”是一个干扰因子。因此，服用与不服用阿斯匹林的人群不是随机分布的，从图中看来，这甚至导致了因果关系推断的方向都发生了改变。

这里如果你的目的是预测，那么一条如图 11-3 那样的拟合曲线完全可以满足预测的要求，而且预测效果可能还相当不错。当我们的任务是因果推断时，故事因此发生了根本性的改变，我们不能苟同于那条最佳拟合直线，因为很明显直线的方向与真实因果关系的方向是南辕北辙。

上面的情况在对观察性数据进行回归分析时是极为常见的。如果数据维度很高，你根本不可能知道数据内部到底发生了什么。正如 Madigan 所形容的，面对高维度的数据，“那简直就是西部荒野，我们对其一无所知”。

情况也有可能是这样，在是否“日服阿司匹林”的两个类别中还有性别的分类，如果再按此分类，你可能又会得到与最佳拟合直线一致的结论：服用剂量越大，犯心脏病概率越小。这种分组因素对因果关系推断的方向性的影响称作辛普森悖论，它在因果推断中有着重要的作用，督促人们考虑可能的、会引起因果关系转向的类别因素。

11.5.2 鲁宾因果关系模型

鲁宾因果关系模型是一个数学模型，在观察性研究中用来确认哪部分信息是可知的，哪部分是未知的。

鲁宾模型可以用来回答这一类的问题："我患癌的原因是因为我曾经有烟瘾。"你确定吗？如果确信，你要能够提供证据支持这个论断。另外一个发问的方式是："如果我曾经不抽烟，我就不会得肺癌。"这是一个很有意思的发问方式，但现有的研究方法还不能回答这样的问题。

定义 Z_i 为对个体 i 进行的实验（0 = 控制组，1 = 实验组），$Y_i(1)$ 为实验组的结果（$Z_i = 1$），$Y_i(0)$ 为控制组的结果（$Z_i = 0$）。

我们所关心的个体层面上的因果关系（unit level causal effect）就是 $Y_i(1) - Y_i(0)$。但是，对于 $Y_i(1), Y_i(0)$ 我们不可能同时观测到 Y 可能的两个值。

用一个例子来说明：假设我本人为研究个体 i，如果抽烟则 $Z_i = 1$，否则为 0。如果抽烟并患肺癌，则 $Y_i(1) = 1$，而如果抽烟但未患肺癌，则 $Y_i(1) = 0$。同样的，$Y_i(1)$ 为 1 或 0，取决于我不抽烟时是否患肺癌。抽烟对患肺癌的总体因果影响是 $Y_i(1) - Y_i(0)$。如果确实由于抽烟而患肺癌，它是 1；不管抽不抽烟，如果患肺癌（或者不患）都为 0；如果抽烟但没患肺癌，则为 –1。然而，由于我只可能知道其中一种结果，$Y_i(1) - Y_i(0)$ 的值无法直接计算得到。

在总体的层面，我们知道有多少人为 1。但是从样本的角度来看，我们不能随便将 1 这个值赋给某人（即我们不可能选择某人，使他 / 她主动经历抽烟及癌症）。

这个问题也称作"因果关系推断的基本问题"。

11.5.3 因果关系的可视化

我们可以利用因果关系图来呈现因果关系建模的概念。

首先，用 W 表示所有潜在的干扰因子。这个假设通常是很难站得住脚的，尤其是在流行学的相关研究实例中，潜在干扰因子的个数基本不可能完全确定。关于流行病学研究，会在在下一章做详细介绍。

在关于 Frank 在线约会的例子中，我们找出了一个潜在的扰乱因子（他中意的女性本身是否漂亮）。如果考虑得更加深入，我们当然还可以发现更多的干扰因子，如 Frank 本身是否是个帅哥，或者他最近心情不好，等等。这些因素都会影响到他写邮件的方式（措辞、语调等）以及对方积极回复的可能性。

其次，我们用 A 表示实验的处理。在这里指是时 Frank 是否在搭讪邮件中使用了"漂亮"一词。我们通常假设 A 是二元的（即具有 0/1 的值）。因此，对于 Frank 搭讪的女士，如果 Frank 使用了"漂亮"一词，我们给她赋值 1。但这里需要注意的时，即便 Frank 用"漂亮"形容的是今天的天气，仍给该女士赋值 1。也就是说，我们在这里不考虑"漂亮"一次出现的具体语境，只要在邮件中出现，一律赋值为 1。

用 Y 表示实验的“结果”，它同样是一个二元值，只取 0/1。对“结果”的定义要十分明确。比如我们定义，在 OK Cupid 的平台内，如果 Frank 在发送第一封邮件的一周内收到了该女士的回复，Frank 在邮件中向对方索要了电话号码，并且该女士在回复的邮件中提供了自己的电话号码，我们则将 1 赋给 Y。另外还注意，如果一位女士没看到 Frank 的邮件，但由于其他一些原因还是通过邮件把电话号码告知了 Frank，我们仍会给 Y 赋值 1。

因果关系图中的干扰因子，实验的“处理”和实验的“结果”都用节点表示；用箭头表示因果关系的方向。换句话说，箭头的出发节点是因果关系的“因”，而所指向的是“果”。

Frank 的例子可以表示为图 11-5。

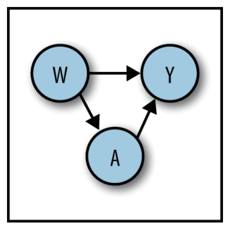


图 11-5：一个实验“处理”，一个干扰因子和一个实验“结果”的因果关系图，对应 Frank 的例子

在 OK Cupid 的案例中，因果关系图十分简单，只包含一个实验“处理”，一个干扰因子及一个实验“结果”。在实际问题中，因果图会变得非常复杂。

11.5.4　定义：因果关系

假设实验共有 100 个人，实验的处理是服用“某种药物”，实验的结果是“患癌”与否。实验的最终结果显示共有 30 人患癌，这意味着患癌率为 0.3。因果推断想要回答的问题是：服用该药物是否导致了罹患该癌症？

为了回答这样一个因果推断的问题，从逻辑上来说，我们需要知道，如果没有服用该药物，会有多少人患癌。假设对于同样的 100 个人的群体，我们规定他们不服用该药物（也许我们是上帝，但即便上帝这么做也是不道德的，人都有生命健康权）。最终的结果是，患癌的总人数减少到 20 人，患癌率为 0.2。我们用两种情形下患癌率的差值表示实验“处理”对实验“结果”的*因果影响值*。在这里，因果关系的影响值为 $0.3 - 0.2 = 0.1$。

> 有时候因果关系的影响值也可以用比率值表示，而不一定是差值。

然而，我们并非上帝，不能让同一批人接受药物“处理”的同时还可以用作控制组。现实

的情况是，他们要么在实验组，要么在控制组。因此，我们需要一个独立的控制组，在该组内的人们不服用此药物。假设这个控制组中人们患癌的概率为 0.1（可以理解为人们自然患癌概率）。那么，通过它，我们可以计算出由于服用该药物而导致的癌症率实际上提高了 20%。但这个结论往往是站不住脚的，因为实验组与控制组总有一些不同质的地方我们没有考虑到[1]。

如果实验组和控制组的两批人是完全同质的（这在生物学上基本是不可能的），所有的干扰因子都可以被排除。但是，就像我们说过的，也许上帝可以做到，但是我们做不到。

因此，对于观察性数据来说，最重要的莫过于确定如何把样本分成实验组和控制组。倾向性得分匹配法就是最为著名的方法。从本质上来讲，倾向性得分法是一种伪随机分组法（pseudo-random experiment），人为控制组的分类原则是尽量选择与实验组总体特征特别相像的个体（这些个体以同样的概率出现在实验组和控制组中）。具体该怎么做呢？注意前面提到的“特别相像”的概念，有很多方法可以找到相像的个体，逻辑回归是其中较为常见的方法。

倾向性得分匹配法分为两个步骤。第一步是用逻辑回归计算每个人接受实验“处理”的概率；然后我们将接受实验与未接受实验的人匹配起来。其实，被分在“伪控制组”的个体具有同等的可能性被分在“伪实验组”，只是实际上没有而已。第二步就是假设样本已经被分成了实验组与控制组，可以应用常规随机实验的分析方法，分析分组之后的数据。

例如，如果我们想要研究抽烟对患肺癌的影响，由于观察性研究的限制，我们需要尽量找出（或者观察到）有同样抽烟可能性的人。因此，能够搜集到的个人信息越多越好：包括性别、年龄、父母是否抽烟、配偶是否抽烟、体重、饮食习惯、运动习惯、一周工作的小时数、血液检测结果等。第一步的逻辑回归模型，Y 变量是个体是否抽烟。逻辑回归模型可以根据个体的体征输出个体抽烟的概率值。这些概率值就是每个个体的倾向性得分，得分值的作用是为了匹配样本，以尽量保证分组的同质性。这里我们假设所有的个体特征变量都是可以观测到的，但事实上，对某些变量的观测和样本收集会比较困难。

这也是倾向性得分匹配的先天性缺陷：我们永远无法确信已经考虑到了所有该考虑的因素。然而，它的优越性在于，如果可能的干扰因子都能被考虑到，那么在倾向性评分匹配模型背景下的因果关系推断是合理和有效的。

复杂的数据和复杂的因果推断问题，也对应复杂的倾向性得分匹配方案。一些关于倾向性得分模型的程序包也设计得很好，很多匹配都可以自动化地完成，使用者不需要关注太多细节。当然，使用何种模型计算倾向性得分，以及该模型中的 Y 变量（要与相应的因果推断对应）还是需要使用者自行设定的，除此之外的事情都很具体化。

注 1：指的就是干扰因子的影响。

在之前的约会例子中，我们需要用什么样的数据来估计因果关系呢？一种办法是用类似土耳其机器人这样的工具，人工浏览所有收到 Frank 邮件的女士的资料，并且将长得漂亮的标记出来。这样我们就可以直接推断收件人本身是否“漂亮”这个扰乱因子对分析结果的影响。这在因果推断研究中叫作分层法，下一章会详细论及。该方法虽然有效，但是也会带来不少问题。

11.6 三个小建议

关于建模，Ori 与大家分享了三个小建议。

第一，当进行因果推断时，深入地了解数据的生成过程至关重要。因为任何模型都会有相应的模型假设，数据本身的数据生成过程可能会明显背离这些假设。如果假设明显不符合数据的生成模型，那模型的使用就应该打上问号。

第二，数据分析的第一步应该置身数据之外，弄清楚到底想要分析的问题是什么。可以把问题写下来，这会帮助你思考，然后再一步一步地思考使用什么样的工具解决这些问题。在使用工具分析数据的过程中，要不时地回头想想当初想要回答的问题，以及正在做的事情是不是在正确的轨道上。这听起来很有道理，也稀松平常，但人们往往都会忘了这么做，忘记了自己分析问题的初衷。

最后，当你运用算法分析数据时，不要被算法和代码冲昏了头脑。不要以为只要算法收敛，模型参数估计显著就一切大吉了。在数据分析时，要时刻保持一颗清醒的头脑，人脑可以发现电脑所不能发现的逻辑性的、常识性的错误；人也应该在数据分析中扮演主导型的角色。

第 12 章

流行病学

本章的贡献者是 David Madigan 教授，他是哥伦比亚大学统计学院院长。Madigan 的研究领域包括贝叶斯统计、文本分析、蒙特卡洛模拟方法、药物警戒系统、概率图模型等。在这些领域，他发表了不下于 100 篇学术文章。

12.1 Madigan的学术背景

Madigan 1980 年毕业于都柏林三一学院（Trinity College Dublin）。他本科主修数学，但在最后一学年选修了一些统计学的课程，并且自学了一些关于计算机的知识，包括 Pascal 语言、操作系统、编译器、人工智能、数据库理论等。大学毕业后，Madigan 先后就职于一家保险公司和一家软件公司，前前后后干了 6 年。期间他的主要工作是对专家系统（expert system）进行研究。

那个时候，个人计算机还没有问世，编程都是通过脚本语言在大型计算机上实现的。Madigan 在保险公司工作的时候，主要从事保险产品定价策略方面的编程工作。另外，他之前还做过一个污水处理系统的项目，并且学了一些可视化方面的知识。他知道如何利用计算机上的显卡进行编程，但是对于数据，他那个时候还接触得很少。

工作了几年之后，他回到了都柏林三一学院完成了博士学位。博士毕业之后，他选择进入学术界，并且在华盛顿大学获得了终身教授的职位。那个时候，机器学习和数据挖掘方兴未艾，他很快就对这两个研究方向产生了强烈的兴趣。他之前还担任过 KDD 数据竞赛的大会主席。此间他学会了使用 C、Java、R 和 S+ 编程。但是即便是当了教授之后，他还是很少和实际的数据打交道。

他说，在刚开始他是一个典型的学者：知道如何编程，却在接触到一个大型的医疗数据时不知道从何做起。因为当时这个医疗数据包括 50 个从不同数据库中收集的不同格式的数据，面对这样的数据 Madigan 当时一筹莫展。

在 2000 年的时候，他去 AT&T 实验室工作了一段时间。据他描述，AT&T 实验室的研究环境是完全学术化的，他在那里学习了 Perl 语言、awk 语言和一些 Unix 的基本知识。他甚至还做了很多网络检索方面的工作。

在那之后，他选择了自己创业——和朋友一起成立了一家互联网公司。该公司的主要产品是一款消费者活动实时可视化系统。

创业的经历给了 Madigan 很多分析大型医疗数据的机会。他曾在不少医疗纠纷的审讯中出庭作证，向法官提供医疗实验数据的分析结果。他说，做这些工作让他在数据解释的问题上大开眼界："把逻辑回归解释给法官听，比我在这里给你们讲课要还要难上加难。那是一种完全不同的、全新的挑战。"Madigan 觉得，简洁明了的可视化对解释分析结果确实有很大帮助。

12.2 思维实验

假设我们手头有一套详细的医疗诊断数据。这套数据是关于每个人的医疗历史的详细记录（统计学中称作纵向数据）。样本量大约为 8000 万个病人，数据记录的内容包括每位病人的处方药单、每次看病的检查结果、每一次医院或者医生家访的检查结果、手术的数据等。每条记录发生的时间都有详细的记录。面对这样庞大的数据，我们能干些什么呢？

从现实情况来看，我们还在延续着中世纪以来的传统，医生对这些数据充耳不闻，看病只依赖于自己的主观判断。但是我们是否能够做得更好一些呢？这些数据能否带来更好地医疗保健体验呢？

这其实是一个非常重要的全民医疗问题。而对于医疗保险公司来说，充分合理地利用好这样的数据至关重要。一个感兴趣的方向是，如果在需要住院之前，医生能够及时地介入病人的治疗，这必将节省大量的医疗资源。要实现这个目标，是需要大数据做分析支撑的。

Kaggle 之前有一个关于医疗数据的分析竞赛，竞赛的题目叫作"Improve Healthcare, Win $3 000 000"（"改善医疗保健，赢得 300 万美元"）。该竞赛的题目要求参赛者能够准确地预测人们在第二年是否需要看病。当然，由于隐私问题，实际竞赛数据中关于个人隐私的数据都被舍去了。

涉及公民个人隐私的数据如果被非法使用，会产生相当严重的后果。比如说，不法分子可以从数据中找到那些有钱的病人，并想方设法敲诈他们。不良的保险公司可能会利用这个数据找到那些医疗风险大的人，取消与他们的保险合约。这些既是法律问题，又是道德问题。在涉及大量个人隐私信息的数据的使用上，我们既要恪守道德标准，也不能触犯法律的红线。

12.3 统计学在现代

想象一下，就在 20 年前，统计学研究在学术界是一番怎样的光景？统计学家要么坐在办公室里证明各种各样的定理，要么在构思某种新的假设检验的方法，或者是缺失值处理的方法。他们根本不会、也不需要跟数据打交道。学术界的统计学与实际的数据分析是脱节的，统计学也不需要领域专家的帮助，统计学家就只沉浸在自己的小世界里。

现在的统计学研究已经发生了本质的变化。顶级的统计学杂志开始看重统计在实际问题上的应用。如果论文是社会科学家（或者很多其他应用科学家们）与统计学家一起合作的成果，往往都很受欢迎。Madigan 也指出，在医学研究领域，统计学家正在扮演越来越重要的角色。

Madigan 觉得现在的机器学习在学术界的发展情况十分类似于 20 年前的统计学。机器学习是一个新的学术研究领域，学术会议的发展也已经比较成熟，但是作为机器学习界的一员，Madigan 还是觉得它仍未脱离当初统计学家们闭门造车的风格：大家都在绞尽脑汁开发新的算法，并千方百计地找数据验证这些算法。Madigan 觉得，机器学习的发展如果不与其他应用领域结合起来，很难有长足的进步。

Madigan 指出，现在很多的统计学家都不注重提高自己解决实际问题的能力，然而现实世界的发展已经对统计学家提出了更为实际的新要求。（当然，Madigan 的同事 Mark Hansen 是一个明显的反例）。

12.4 医学文献与观察性研究

观察性研究的内容我们在上一章已经讨论过，它是统计学中的一个十分重要的研究方法，也是医学研究的标准方法之一。观察性研究的结果对于医学培训、临床以及政府的医药智力起着举足轻重的作用。

比如说，Jane Green 等人有一篇合著的论文，题为“口服磷酸双酯提高患食管癌、胃癌和大肠直肠癌的风险：基于英国初级医保数据的病例对照分析研究”（参见 http://1.usa.gov/16UfNjZ）。Madigan 看完这篇论文后总结说：这篇论文处理的问题与当初对阿司匹林的研究一样，就是干扰因子的问题。该论文的结论是，口服双膦酸盐显著地提高了患这些癌症的风险，提高的幅度起码达到了 10%。

这篇文章刊登在《纽约时报》某期的首页，研究者是几个没有利益冲突的学者[1]，研究的样本来自数以百万计的患者服药数据。但即便是这样，其研究结论也很可能是错误的，并且之后的研究成果也很可能推翻这一结论。

注 1：没有利益冲突代表作者互相之间没有利益瓜葛，既没有侵权也没有利益瓜分之嫌，因此研究的结果较为可信。

这样的例子还有很多很多。这其中是一个没有引起人们足够重视，却又十分重要的问题。

政府每年在医药研究上都会花去大笔的金钱，因为医药研究的结论会关系着无数人的生命健康。因此，我们必须在正确的理论指导下，做严肃的医药研究。

12.5 分层法不解决干扰因子的问题

流行病学研究是一个与干扰因子做斗争的过程。但是可惜的是，现有的流行病统计学的研究方法还不能很好地解决这个问题。其中最常使用的方法是*分层法*。举个例子来说，如果我们觉得性别是可能的干扰因子，在研究的时候就按照性别把样本分成两层，并在分析的时候根据不同性别样本的权重调整估计值的结果。

然而，当实验的结果中数值比较小，或者不同层总体之间差异显著时，分层法会影响因果分析的有效性。

表 12-1 是某实验结果的汇总表，我们用这个表举一个例子。

表12-1：某实验汇总表

	实验组：已服药	实验组：反事实	对照组：反事实	对照组：未服药
Y=1	30	20	30	20
Y=0	70	80	70	80
P (Y=1)	0.3	0.2	0.3	0.2

从表中可以得出，该实验的实验组和对照组的人数均为 100 人，表中间的两列实际上是不可观测的。由汇总表的结果可以得到，因果关系效果值为 0.3 – 0.2 = 0.1，也就是 10%。

然而，当我们按照性别将汇总表成两个子表之后，问题就来了，尤其是当这些表中出现很多很小的数目的时候。分层之后的男性子表见表 12-2，女性子表见表 12-3。

表12-2：分层结果：男性子表

	实验组：已服药	实验组：反事实	对照组：反事实	对照组：未服药
Y=1	15	2	5	5
Y=0	35	8	65	15
P (Y=1)	0.3	0.2	0.07	0.25

表12-3：分层结果：女性子表

	实验组：已服药	实验组：反事实	对照组：反事实	对照组：未服药
Y=1	15	18	25	15
Y=0	35	72	5	65
P (Y=1)	0.3	0.2	0.83	0.1875

由表 12-2 得知，男性的因果关系效果值为 0.3 – 0.25 = 0.05，而表 12-3 告诉我们相应女性的因果关系效果值为 0.3 – 0.187 5 = 0.112 5。最后的研究报告可能会宣称该药物对女性的效果是男性的两倍多。

换句话说，分层的想法有时候不仅没有解决问题，反而带来了很多新的问题：因果关系的估计结果往往变得扑朔迷离。因此，当你在解决干扰因子问题时，至于到底该不该用分层法，需三思而后行。

人们在实证中到底如何处理干扰因子的问题

虽然说分层法存在一些潜在的问题，但是在实际分析中，分层法仍然是使用最多、应用最广泛的方法。对于明显的或潜在的干扰因子，都可以针对该因子变量将样本分层再做分析，或者针对该因子做一些模型层面的调整（比如上一章详细讨论过的“倾向评分匹配法”）。因此，如果我们觉得某实验中，服用阿司匹林是一个干扰因子，那么可以针对该因子将样本分层以排除该因子对因果关系推断的影响。

这里有一个有趣的例子（http://goo.gl/3VgRi0）：某项实验的目的是研究口服避孕药与静脉血栓的关系。研究人员在实验中考虑了一些他们认为可能的干扰因子，并得出了以下结论：

> 在调整了人们服药长度的可能影响之后，实验的结果显示口服避孕药的女性患静脉血栓的几率是不服用口服避孕药女性的两倍。

该研究的结果曾经轰动一时，美国广播公司对此也有过报道。但是，该实验的研究人员在考虑干扰因子的时候显得有些不走寻常路。按理说，是否服用其他药物，比如阿司匹林这样的常用药，应该是一个明显的干扰因子。因此，即便分层法可以在一定程度上解决干扰因子的问题，但是应该如何选取干扰因子却是一个令人头疼的问题。另外一项研究中，研究人员将实验对象的静脉血栓史作为干扰因子，得到了完全不同的结果。

这个例子告诉我们，干扰因子的选择对实验结果有着直接影响，选择不同的干扰因子，得出的结果可能有天壤之别。譬如之前的口服磷酸双酯的研究课题，研究人员选择了吸烟与否、饮酒与否以及 BMI 这样的干扰因子。但是如果另外一个研究选取另一批干扰因子，得到的结果可能大相径庭。从这一点来看，处理干扰因子是一个无解的问题，我们几乎不可能把所有的干扰因子都考虑进来。这是一个哲学问题，而不是一个科学问题。

Madigan 和几位合作者曾经做过一项研究，他们让一批流行病学家跟对同一组课题设计 5 个实验。最后发现，这些设计在底层环节上都非常一致，但是对于考虑什么样的干扰因子，每个人都自诩专家，并宣称自己的设计是最合理的。

12.6 就没有更好的办法吗

我们无法直接回答这个问题，但是可以给你一个很醒脑的例子。Madigan 和他的合著者做过另外一项研究，他们选取了 50 个有关药物及其副作用的课题（比如抗生素与消化道内出血的关系等）。每个课题都放在 9 个数据集上做接近 5000 个不同的验证分析。

比如，血管张力素转化酶抑制剂会引起心悸，针对此课题他们在 9 个数据库上做了接近 5000 个验证分析。最小的数据库有 400 万个病人的数据，而最大的一个数据库样本量达到了 800 万人。

针对这一个例子，某一个数据库的分析结果显示血管张力素转化酶抑制剂引起心悸的几率是对照组的 3 倍，而另一个数据库的分析结果显示是 6 倍。这里虽然研究的结果在不同数据库上有所不同，但起码它们的指向都十分一致，那就是该药物确实增加了心悸发生的几率。

然而，对于 50 个课题中的 20 个课题，在不同数据库上得到分析结果似乎完全是随机的。有些具有统计显著性，有些没有。有些具有正的显著性，有些具有负的显著性。也就是说，你能想到的结果都能在某个数据库的某项验证分析结果里找到对应的。图 12-1 展示了这 50 个课题的统计显著性分布。刚才关于心悸的课题位于该图的最上方。

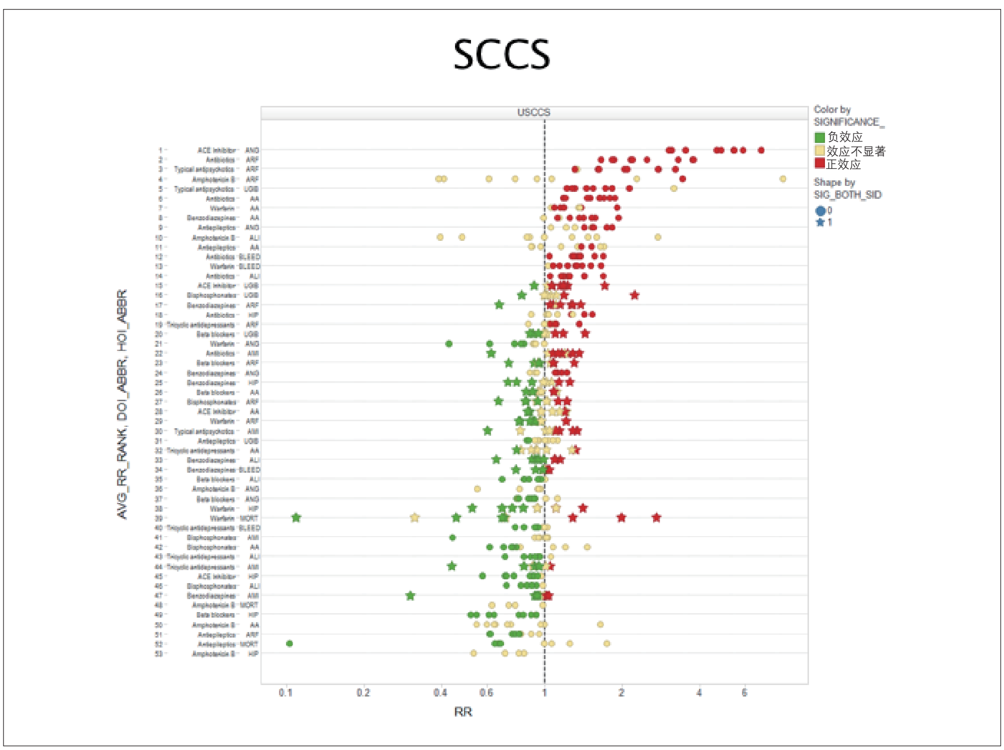


图 12-1：50 个课题的统计显著性分布图（另见彩插图 12-1）

数据库的选取对研究的结果有直接影响，但是现在的大多数论文都对此条避而不谈。

Madigan 团队接下的研究更加有说服力。他们详细地考虑了每个实验的性质和研究的可能性，只要有模棱两可的问题，那么所有的可能性都会考虑进来。具体说来，他们考虑了这样一些可能性：使用哪个数据库；考虑哪些干扰因子；检查的时间窗长度（比如一个心脏病人在停药后一周发病的概率和一个月发病的概率明显会有所差异）等。

他们得到的结论是：所有的实验结果都可能有两个对立的解释！

回到刚才口服磷酸双酯的例子。一团队的论文（http://1.usa.gov/16UfNjZ）宣称口服磷酸双酯会增加患癌的风险，而另外一篇发表于 JAMA 的论文（http://1.usa.gov/1hi2kbj）则表示口服磷酸双酯不会增加患食管癌的风险。他们分析的数据甚至来自于同一个数据库，却得到了截然相反的结论。其实这样的研究成果具有对立性的现象在医学界还有很多。

12.7 研究性实验（OMOP）

为了直接解决上面所提的这些问题，或者说起码在一定程度上突破现有研究理论和方法的瓶颈，Madigan 加入了一个叫作 OMOP（http://omop.org/）的研究项目并担任了首席研究专家。他的使命是开发和实现一些可以用作观测数据分析的新统计方法。这套统计方法是 OMOP 项目的核心贡献。OMOP 的全称是“Observational Medical Outcomes Partnership”，我们下面稍作介绍。

关于 OMOP

2007 年的时候，由于认识到电子健康信息（EHR）数据和其他大型的公平健康数据规模正井喷式地增长，这样的数据为医药和人体健康方面的研究提供了宝贵的契机，国会决定以 FDA 牵头，打造一个新的医药监控项目，以更好地保护公民的健康安全。FDA 随后牵头了一些项目，包括非常著名的 Sentinerl 项目。该项目旨在搭建一个全国性的数据网络。

在该背景下，国家健康委员为与 PhRMA 以及 FDA 发动了 OMOP 项目。该项目是典型的公私合作模式。OMOP 项目已经为业界带来了一些令人瞩目的研究成果，其中的一项研究成果确认了研究和分析超大型同质性数据的流程和方法。

OMOP 项目的贡献者来自于流行病学、统计学、计算科学以及许多别的学科。他们通力合作，只为了回答下面这些问题：医药研究者能从这些大型健康数据中得到哪些有用的信息？有没有一个普适的流程和方法可以用来研究和分析不同数据库的数据？研究的结果能不能被很好地验证？

如果对上述问题的回答都是肯定的，那么这将为医药和健康研究领域带来翻天覆地的变化。从长远来看，整个国家的医疗系统以及医疗监管系统也将发生质的变化，人们的健康安全将得到更好的保障。

Madigan 和他得团队选取了 10 个大型医疗数据库，这些数据库来自于保险公司和 EHR 等机构，涵盖了大约两亿人的资料。这绝对是一个大数据！

他们详细地研究了观察性研究中经常使用的模型，并用上面的大数据进行了一一验证。最后他们选取了 14 个常用的纵向数据流行病学模型。模型的细节被全部自动化，并且每个模型都有大约 5000 个可调参数。

这样做是为了验证已有模型的预测效果是否准确。

因为公认的研究成果可以用来验证这些模型的效果。他们选取了 10 个被广泛研究的药物，包括血管素转换酶抑制剂、华法令阻滞剂以及苄丙酮香豆素等；以及 10 个用药症状，包括肾功能衰竭、住院和出血症状等。

对于这 10 类药物来说，其中某些的副作用是已知的。比如说，华法令阻滞剂会导致血管壁变薄以及失血。类似这样的已知的副作用还有多达 9 个。

另外有 44 个已知的无副作用药物，这些药物我们有充分的证据相信他们不会对人体产生副作用，最起码不会产生上述 10 种用药症状。

验证的方法很简单：在所有 10 个数据库数据上运行 5000 种常见的流行病学分析，并验证每个数据库上每种分析的预测效果。这有点类似于第 4 章讨论过的垃圾邮件过滤模型的测试问题，数据被分为训练数据集和测试数据集。模型的估计只用到训练数据集，而测试则是在不同于训练数据集的独立的测试数据集上完成。

每次测试都只输出两个统计量：相对风险值（Relarive Risk，RR）[2] 和预测误差。

可以看出，这个项目的最终目的是验证已有流行病学模型的实际预测效果，这有点类似于 John Ioannidis 的工作（http://stanford.io/15LfJDL）。

为什么这个项目到现在才做？

这涉及利益冲突的问题。没有人愿意去验证他们的模型是无效的，他们自然也不会提供数据给想要验证的人。而且从规模上来看，这个项目是巨大的，其耗资近 2500 万美元。然而，这笔钱看起来似乎很多，但是与那些已经完成的（结果可能是错误的）研究的经费比起来，这简直就是九牛一毛。

注 2：相对风险值是因果关系的测度指标

这个项目购买了 10 个数据库中的所有数据，所有的模型都被自动化，他们还使用了亚马逊的云计算服务用来加快模型的运行速度。他们甚至开源了该项目所有的原始代码。在项目的第二阶段，他们把研究对象缩减到了 4 个用药症状，并且绘制了所有模型汇总的 ROC 曲线（这意味着这条曲线价值 2500 万美元！），见图 12-2。

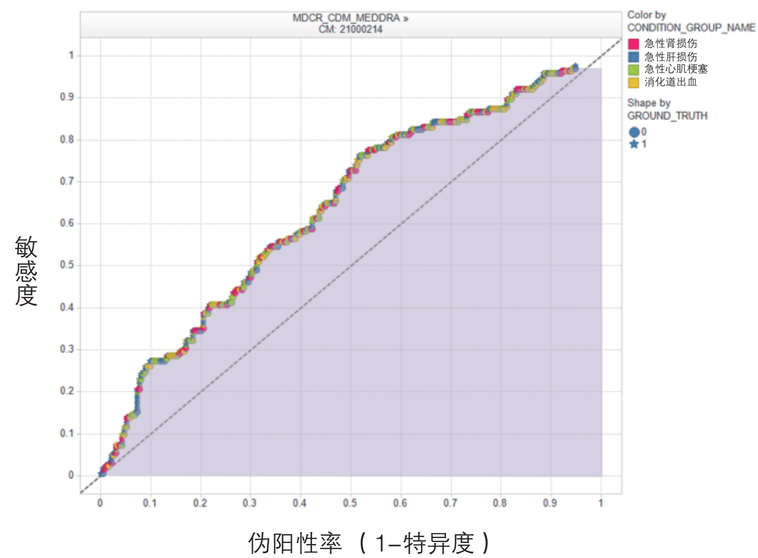


图 12-2：价值 2500 万美元的 ROC 曲线图（另见彩插图 12-2）

为了理解这幅价值连城的 ROC 图，我们首先需要定义一个阈值，比如说 2。也就是说，如果模型的相对风险值大于 2，则模型所估计的因果关系可以解释为“恶化作用”，如果小于 2，则为“促进作用”。阈值的选取对模型结果的解释有着直接的影响。

如果阈值设定为 10，那么没有任何一个模型的相对风险值可以达到 10，因此所有的模型结果都是有“促进效果”。另外需要注意的是，由于一些药物已经退市或者停产，他们的市场保有量为 0 或者极少，因此模型的“敏感性”自然很低，因此模型很难发现有任何效果。

当然，低“敏感性”也意味着伪阳性的值也很小。

同样的道理，如果阈值设定为 –10，那么所有模型的相对风险值都大于 –10，所有模型得到的都是“恶化作用”。此时虽然模型的敏感性”达到了 100%，但是伪阳性的值却变大了。

因此，阈值的设定相当于在模型的“敏感性”和伪阳性中做出权衡，这就是 ROC 曲线的核心内容。我们可以找到一个最佳的阈值 1.8，其“敏感性”为 50%，伪阳性为 30%。

如果你是 FDA，这样的 ROC 曲线是不能过关的。因为 30% 的伪阳性不在 FDA 的考虑范围之内，会被立即否决。

在前面的章节我们讨论过，相对于 ROC 本身，可以用 AUC 来评价模型的预测效果。AUC 为 1 的模型是最佳模型，如果模型的预测效果十分平庸，则 AUC 接近于 0.5。0.5 的 AUC 代表该模型与瞎猜没有任何区别。

上图的 AUC 值为 0.64。在研究团队（包括 David Madigan）运行过的 5000 个分析中，0.64 其实是其中效果最好的模型。

但是需要注意的是，0.64 的 AUC 是建立在对每个数据都应用同一个模型的情况。可以想象，现实中使用的很多模型也基本与瞎猜没有任何区别。

然而，没有任何一个流行病学家会这么干。对于手中的数据，他们总是想找到一个最好的模型以便得到最好的预测效果。他们通常都能得到更好的结果，以上面医疗保健的数据为例，在预测急性肾损伤时，他们使用的模型的 AUC 达到了 0.92（见图 12-3），模型的“敏感性”为 80%，伪阳性值为 10%。

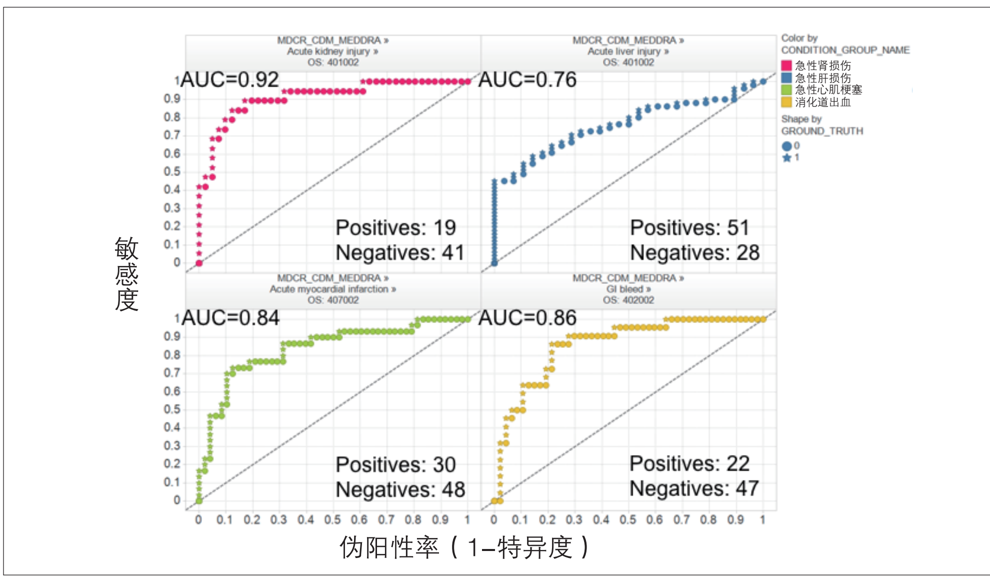


图 12-3：针对性的选取模型后，模型的预测效果可以得到较大改善（另见彩插图 12-3）

模型的验证使用了交叉验证法。最终的最优模型是一个叫作“OS”的方法，该方法在给定病人的病历后，在病历期内进行比较（因此，病人在服药阶段和不服药阶段会有较大差距）。不过该方法还没有引起大家的注意。

有意思的是，大多数流行病学家根本不承认这项研究的结果！

如果感兴趣，你可以去该项目的官方网站（http://elmo.omop.org）看看，上面有每个数据以及每个模型的 AUC 曲线。模型使用的数据大致截止于 2013 年中。如果有最新的数据，更新所有的分析，结果可能会有所不同。

12.8 最后的思维实验

在上面的 OMOP 项目中，每个数据库都运行了近 5000 种分析。有没有可能将这些分析捆绑起来做呢？或者说，用一种类似模型投票的方式，赋予模型不同的权重以达到更好的分析效果。因为该项目的源代码是开源的，说不定你可以就这个想一想题目写一篇博士论文呢。

第 13 章

从竞赛中学到的：数据泄漏和模型评价

本章由 Claudia Perlich 贡献，过去几年，她一直担任 M6D 公司的首席科学家。在这之前，她供职于 IBM 某研发中心的数据科学小组，在电视节目 *Jeopardy*! 上击败人类赢得比赛的 Waston 系统就诞生于这个研发中心，但是 Claudia 并没有参与那个项目。Claudia 拥有计算机科学硕士学位，并在纽约大学获得信息系统博士学位。现在她在商学院开设了一门数据科学课程，课程的重点是如何评估数据科学工作的价值和如何管理数据科学家。

Claudia 蜚声数据科学界，因她在数据挖掘竞赛中是常胜将军。她曾经赢得过 2003 年、2007 年、2008 年和 2009 年的 KDD Cup 竞赛，还赢得过 2005 年的 ILP Challenge、2008 年的 INFORMS Challenge 和 2010 年的 Kaggle HIV 比赛。

近些年 Claudia 转而成为这些赛事的组织者。她先是作为组织者参与了 2009 年的 INFORMS Challenge，继而又参与了 2011 年的 Heritage Health Prize。最近 Claudia 宣布不再参与数据挖掘竞赛，但我们的课程有幸请她来，跟大家分享一些经验见解，看看我们能从数据竞赛中学到什么。在数次竞赛中，Claudia 获益良多，特别是数据泄漏和如何评价她在竞赛中提出的模型。

13.1 Claudia作为数据科学家的知识结构

Claudia 先询问了其他数据科学家的参考点，根据其知识结构来评估他们在数据科学界所处的位置。（她的知识结构如表 13-1 所示。对比了第 1 章讲到的数据科学家的知识结构，

Claudia 说：“有一项最重要、最难形容的技能没有出现在你们的知识结构中，那就是*数据*。”）她认识一些世界顶级的数学家、机器学习专家和统计学家等。在知识结构表各项技能的描述中，她是以专家的标准来要求自己，还是以她所处领域的平均水平，或者只是以一个普通人作为参考？

表13-1：Claudia的数据科学知识结构

	及格	优秀	扎实	注　释
可视化	×			我会做可视化，但是我不相信可视化
计算机科学	×			我有两个计算机科学专业的硕士学位，我也可以随手编写一些代码，但不是那种产品级的代码
数学	×			学数学是很早以前的事了
统计学		×		没接受过正规训练，很多东西都是边做边学，有时候也要靠自己很好的直觉
机器学习			×	
领域知识				你确信这个问题没问错吗
讲演			×	
数据			×	

13.1.1　首席数据科学家的生活

过去，Claudia 花了很多时间和精力在预测建模上，包括参加一些数据挖掘比赛，为一些学术期刊、学术会议（比如 KDD）撰写论文、做演讲、写专利，在大学教书等。但她最喜欢干的事是挖掘数据的含义，她喜欢直接面对数据，经由数据了解世界。

Claudia 有 15 年的数据处理经验，期间她通过深入研究*数据的生成过程*，培养出一种对数据的直觉。数据生成过程是整个研究问题中的关键一环。她花了大量时间来思考模型评价过程，从而也培养出了对模型的直觉。

Claudia 的主要技能包括使用 Unix、sed、awk、Perl 和 SQL 对数据进行处理。她能使用各种方法进行建模，包括逻辑回归、*k* 最小近邻算法等，最重要的是她花了很多时间将所有的事情都串起来。整理数据、为数据进行建模大约花去她 40% 的时间，这时她将自己称作“贡献者”。她还要花 40% 的时间写文章、做演讲，大多数时候是代表 M6D 公司和外界进行交流，这时她称自己为“大使”。还有 20% 的时间用来与她的数据科学团队一起工作，这时她的角色则是“领导”。

13.1.2　作为一名女数据科学家

女性在数据科学领域同样可以工作得很出色，她们的直觉在这里是有用的，并且常常被用到。当事情不对劲的时候，她们可以嗅出空气中异样的气息，当然这和使用算法严格来进行证明是两回事。通常，人们更容易记住女性，即使该女性不一定能记住他。Claudia 愉快

地承认了这个事实。但是，她之所以有现在这样的成就，根本原因还是她很优秀。

在学术界，Claudia 曾发表过不少文章，因此熟知在期刊之类的地方发文章的流程。她讨论了在学术期刊或者学术会议上发文章时，两性是否平等。在过去，这是一个双盲的过程，但是现在更多是单向的。而且根据 Shawndra Hill 和 Foster Provost 在 2003 年发表的一篇文章，根据文章的引用文献就能猜测出文章的作者，而且准确度达到 40%，如果该作者发表过多篇文章，预测准确率还会更高。希望这个方法不会在审阅文章时被用到，这只是为了说明，盲审不一定能解决问题。最近，文章开始允许署名，希望这一举动不会造成过多的偏见。Claudia 承认自己在一些研究机构中曾遭遇过少许偏见——根据她的经验，某些研究机构的准备工作做得更好。

13.2 数据挖掘竞赛

Claudia 对于不同类型的数据挖掘竞赛做了总结。第一种类型是“无菌型”的。数据是事先准备好的、干净的，对错误的衡量也有统一的标准，特征变量常常是匿名的，这是一个纯粹的机器学习问题。

这种类型竞赛的典型例子是 KDD Cup 2009 和 Netflix Prize，还有一些 Kaggle 竞赛。在这种比赛中，重点是算法和计算。获胜者通常组合了一堆复杂的模型，调用了庞大的计算资源。

KDD 杯数据分析竞赛

历届 KDD 杯数据分析竞赛的题目和对应的数据集都可以从以下网站获得：http://www.kdd.org/kddcup/index.php。下面是历届比赛的题目：

- KDD Cup 2010：Student performance evaluation
- KDD Cup 2009：Customer relationship prediction
- KDD Cup 2008：Breast cancer
- KDD Cup 2007：Consumer recommendations
- KDD Cup 2006：Pulmonary embolisms detection from image data
- KDD Cup 2005：Internet user search query categorization
- KDD Cup 2004：Particle physics; plus protein homology prediction
- KDD Cup 2003：Network mining and usage log analysis
- KDD Cup 2002：BioMed document; plus gene role classification
- KDD Cup 2001：Molecular bioactivity; plus protein locale prediction
- KDD Cup 2000：Online retailer website clickstream analysis
- KDD Cup 1999：Computer network intrusion detection
- KDD Cup 1998：Direct marketing for profit optimization
- KDD Cup 1997：Direct marketing for lift curve optimization

与之相反，另一种是“现实世界”中的数据挖掘竞赛。面对的是原始数据（数据经常分散在不同的表中，而且合并起来很困难），需要自己建立模型，根据任务特性评价模型的好坏。这种类型的比赛更好地模拟了现实世界，这就回到了 Rachel 在本书前面的一个思维实验：如何在课堂上模拟一个数据科学家所要面对的混乱局面。这需要在长期处理混乱情况的过程中不断历练。

这种类型的例子有 2007 年、2008 年和 2010 年的 KDD Cup 比赛。如果你参加的是这种类型的比赛，那么就需要了解问题所处的领域，分析数据，建立模型。获胜者会是那些最懂得如何根据实际问题对模型进行调整的人。

Claudia 更喜欢第二种类型的竞赛，因为这和我们的生活贴得更近。

13.3 如何成为出色的建模者

Claudia 认为，数据和领域知识是数据科学家所需具备的最重要的技能，但是这些技能是老师教不来的，只能通过不断积累得来。

Claudia 通过参加各种数据挖掘竞赛学到了很多东西，而这些东西在学术界常常被忽视。

- 数据泄漏

  数据泄漏是参赛者最好的朋友，也是组织者和出题者的噩梦。数据总是或多或少有些问题，Claudia 将发现这些问题变成了一种艺术，她总能指出准备竞赛的人在处理数据时的懈怠和马虎。

- 真实世界中评价模型的标准

  标准的模型评价标准有均方误差（MSE）、错分率和曲线下面积（AUC）等，在真实世界中对模型做评价，有时候要越过它们，选择其他标准。比如，利润可能是现实生活中一个更有效的评价标准。

- 特征变量创建和变换

  真实数据很少那么规整（看起来很漂亮的数据），如何解决这个问题现在还是个挑战。

13.4 数据泄漏

在 2011 年的 KDD 大会上，Claudia 和 Shachar Kaufman、Saharon Rosset 共同发表了一篇文章：“Leakage in Data Mining: Formulation, Detection, and Avoidance”（“数据挖据中的数据泄漏：公式化、检测和避免”）。他们提到了另外一个作者 Dorian Pyle，他写过很多关于数据挖掘中数据准备阶段的文章。在这个过程中他发现了一个现象，即很多事情变得不合时宜，他将这称为“时空错乱”。他说很多数据好得难以置信，但其实这种数据的存在本身就是一种漏洞。Claudia 和另外两位作者将预测建模中的类似现象称为“数据泄漏”。Pyle 建议借助探索性数据分析找到数据泄漏的源头，而 Claudia 他们倾向找出对付数据泄漏的方法。

数据泄漏是借助数据或信息做出预测，虽然结果正确，但其实你能够获得这些数据并据此进行预测，这本身就是不合适或者行不通的。不光在比赛中，在真实环境中，这也是建模时的一个大问题。数据泄露通常是因果颠倒的产物，让我们通过几个例子，看看这是怎么发生的。

13.4.1　市场预测

有一场比赛是预测标准普尔指数（S&P）的涨跌的，获胜者的曲线下面积（AUC）达到了惊人的 0.999。股票市场几乎是随机的，出现这种情况要么是因为有人很有钱，要么是因为什么地方出现了错误（提示：那次是有地方出错了）。

在过去一段“美好时期”，参赛者只要找到泄露的数据就能赢得比赛。在这个案例中，虽然我们不清楚到底是哪里产生了数据泄漏，但是通过持续不断地对数据进行分析，有可能就会发现数据集中的有些信息可以对预测标准普尔指数产生决定性作用，但是在真实情况下做预测是无法拿到这些信息的。举这个例子的用意是说明，比赛中有人取得如此高的 AUC 一定是依赖于某些数据泄漏，在真实环境下如此建模是行不通的。

13.4.2　亚马逊案例学习：出手阔绰的顾客

这个比赛的目的是通过历史购物记录，预测出哪些顾客会在亚马逊网站上花更多的钱。数据由不同商品种类的交易记录构成。其中赢得比赛的一个模型将“Free Shipping = True”认定成一个关键的预测变量。请注意，只有购物金额在一定数额之上，比如说 50 美元，才能享受免费送货。

哪里不对劲呢？关键是免费送货是因为花了很多钱的结果。不能因为这样一种相关性去建立模型，比如，这种模型对新顾客就不适用。这里要注意的是，时间戳的作用并不大。包含“Free Shipping = True”的记录和购买行为是同时发生的，用这样的记录做预测是没有意义的。你需要使用以前的数据去预测未来。困难在于收集到的数据中混入了包含免费送货的记录，这些记录必须被人工剔除，而作为模型的建立者，需要深思熟虑，了解你的数据。如果没有把数据泄漏的情况考虑在内，很可能将免费送货作为一个变量来建立模型，并且发现它的预测效果很好。但是，当在生产环境中使用该模型时，你无法知道购物者是否会得到免费送货的待遇。

13.4.3　珠宝抽样问题

还是一个在线零售商的例子。这次是预测哪些顾客会购买珠宝。数据依然由各类商品的交易记录构成。其中一个模型在 sum(revenue) = 0 时，预测结果非常准确。

哪里又不对劲了呢？原来，为这次比赛准备数据的人删除了顾客是否购买珠宝的信息，但数据集中只包含了那些买过东西的用户。因此，那些 sum(revenue) = 0 的顾客，一定是只

买了珠宝的顾客。数据集里只包含有过购买行为的顾客，这本身就是一件很奇怪的事，特别是在顾客未完成购买行为前，你无法使用该数据。这个模型不是在正确的数据上训练出来的，没有任何用处。这是一个抽样问题，也是一个经常碰到的问题。

关于如何对用户进行抽样的警告

上面提到的这个例子，仅对有过购买行为的用户进行分析有点奇怪。你是想要将分析仅局限在这批用户，还是访问网站的所有用户？更一般地，如果你不认真对待手头的用户信息，不把思路理清楚，就有可能犯非常简单但又非常严重的抽样错误。比如想要分析网站一天的用户访问记录，就有可能出现对经常访问网站的用户过采样的问题。

用一个小例子来想一下这个问题：假设有 80 个用户，其中 10 个每天都来访问你的网站，剩下的一周只访问一次，假设他们的访问时间均匀地分布在每周的 7 天里，那么任意一天都会有 20 个用户访问你的网站。这其中的 10 人是每天都来的用户，另外 10 人是每周来一次的用户。这里就会每天对访问网站的用户进行了过采样。他们访问网站的行为也许和其他用户有着根本不同，虽然他们是整个用户数量的 12.5%，但却占据了采样数据的 50%。

13.4.4 IBM客户锁定

在 IBM，需要预测哪些公司有意向购买 websphere 解决方案，数据是一些交易记录，通过网络爬虫抓取潜在公司的主页得来。一个得奖的模型指出，如果 websphere 出现在公司的主页上，那么这家公司购买 websphere 解决方案的可能性就非常大。哪里有错了呢？要记住，潜在客户的定义是那些还没有购买产品的公司（否则，IBM 就不会试图将产品卖给它），因此，没有潜在客户会在它的主页上显示 websphere 相关信息，这根本就不是一个预测变量。如果 IBM 回到 websphere 解决方案还未诞生的年代，能看到网站当时的快照，以此作为数据源，那么这个预测变量是有意义的。遗憾的是，现在的数据已经包含了泄漏的信息，它们已经购买过 websphere 了。再说，你也无法抓取过去的网页，只能抓取现在的网页。

这看起来是一个愚蠢并且显而易见的错误，谁也不会犯这种错误。也许如此，但是这种事时有发生，而且在深入理解数据、弄明白特征和预测变量的含义前，你无法预料到这种事情会发生。想一下，如果这种“显而易见”的错误都会犯，那么对于那些不那么明显的情况更应该加倍小心。同时，这也是在本书中没有给予充分强调的一个例子，与网络爬虫和洋气的机器学习算法相比，通过一些基本的检查，确保一切如你所料，往往会让你走得更远。这可能看起来不够酷，不够吸引人，但它管用，是一个很好的经验。人们不会在聚会时谈起它，也不会将其作为研究成果发布出来，但是它是合理的，而且管用。（不过话又说回来，Claudia 因为使用了这个技巧，赢得了多次比赛，而且经常被邀请参与各种聚会。所以我们收回刚才的话。不，我们不用这样做。关键是把事情做好，其他人自然会跟随你。聚会和名声并不是目标，目标是追求真理。）

13.4.5 乳腺癌检测

假设要研究哪些人患有乳腺癌。看看图 13-1，患者的 ID 看起来无足轻重，却是预测模型中很重要的一个变量。看看发生了什么？

在图 13-1 中，红色代表患有乳腺癌的患者，绿色代表没有患乳腺癌的人，根据患者的 ID 做了一个散点图。很容易发现，病人根据患者 ID 划分成了三四个明显的区块。不同的区块在这里有很强的预测能力。这可能是因为数据来自不同癌症治疗中心的数据库，有些治疗中心专门用于救治那些重病患者——顾名思义，来这个中心就诊的患者得癌症的几率更大。

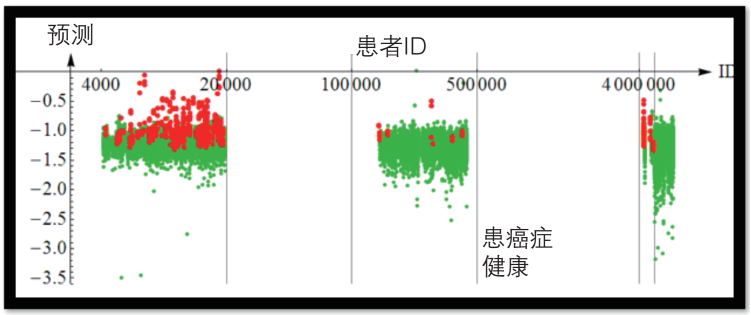


图 13-1：依患者的 ID 排序，红色代表得癌症的患者，绿色代表未得癌症的人（另见彩插图 13-1）

这种情况在课堂上引起了一场有趣的讨论。

> **学生甲**：为了使比赛公平公正，应该给患者重新随机编号。
> Claudia：这能解决问题吗？还会有其他类似属性存在。
> **学生乙**：这取决于除了病人的 ID，还有哪些属性可以用来判断病人来自哪里。
> Claudia：不妨这样想想：我们究竟要使用模型做什么？怎么样才能真正预测出哪些患者患有乳腺癌？

假设现在有一个新来的病人，你将怎么做？如果这个病人的 ID 落在了第五个区块里，那这种根据病人 ID 预测的方式显然不靠谱。但是如果不是这种情况，则使用病人 ID 做预测就是一种很好的方法。

这个讨论将我们带回到一个基本的问题：我们需要知道建立模型的目的、怎么使用模型和模型是否奏效，以帮助我们决定如何构建模型。

13.4.6 预测肺炎

在一次 INFORMS 竞赛中，题目是根据病历预测病人是否患有肺炎——将诊断码当作一个数值型变量时，预测准确率不高（曲线下面积为 0.80），如果将诊断码作为分类变量，则准确率可以提高至 0.90。这是为什么呢？

这和本次比赛的数据准备方式有关，如图 13-2 所示。

icd1x	icd2x	icd3x	icd4x
786	285	459	-1
401	486	-1	-1
401	486 ⟵ 780		-1
599	-1	-1	-1
V22	650	-1	-1
V56	492	586	-1
786	493	285	459

→

icd1x	icd2x	icd3x	icd4x
786	285	459	-1
401	-1	-1	-1
401	780	-1	-1
599	-1	-1	-1
V22	650	-1	-1
V56	492	586	-1
786	493	285	459

图 13-2：INFORMS 竞赛中数据是如何准备的（另见彩插图 13-2）

肺炎的诊断码为 486。所以如果该数据出现了，准备数据时就删掉它（用 –1 取而代之）。数据中的行代表不同患者，列代表各项诊断结果，最多有四项诊断，–1 表示未进行该项目的诊断。

为了避免数据泄漏，在必要时，将其他诊断结果向左进行了平移，这样将 –1 都留在了右边。

这样做有两个问题：

- 如果某一行记录只包含 –1，那么该病人肯定患有肺炎；
- 如果某一行记录不包含 –1，该病人肯定没有得肺炎（除非有五项诊断，但这并不常见）。

仅仅凭借这一信息就能赢得比赛。

发生泄漏

比赛中，利用数据泄漏比构建一个好的数学模型更容易获胜。即使你没有觉察到数据泄漏，你的模型也会感知到这种情况，从而不自觉地利用它取得比赛的胜利。无论怎么说，数据泄漏都是数据挖掘竞赛中面临的大问题。

13.5 如何避免数据泄漏

这里并不是要告诉你如何利用数据泄漏来赢得各种数据竞赛。作为一个数据科学家，在准备数据、清理数据、补偿缺失值和移除异常值等过程中，总会存在数据泄漏的风险。在准备数据时，你可能会无意中扭曲数据，让模型在这份所谓的“干净”的数据上表现良好，但将该模型应用于真实场景中时，效果却糟透了。Claudia 给了我们一些避免数据泄漏的具

体建议。首先需要暂时去掉那些事先已经知道的信息，比如在一个患者确诊前的信息。每一条记录都应该有一个时间戳，记录你得到该信息的时间，而不是这条记录发生的时间。去掉那些制造麻烦的行和列，特别是那些很容易发现的不一致的信息。经过深思熟虑，从一开始就使用干净的原始数据，未尝不是一个很好的实践。最后，你需要知道数据到底是如何产生的。

在前面提到的文章中，Claudia 和她的合作者提出了避免数据泄漏的一个“两步走”方法：首先在收集数据阶段，为所有记录打上合适的标签，然后执行一个他们称作“学习和预测分离”的过程。

13.6 模型评价

怎么评价一个模型好不好？我们已经在前面一些章节中讨论过这个问题，但是，从专家那里听取一些建议总是一件好事。

在使用强大的算法寻找模型中的模式时，存在过拟合的风险。这个概念有点难理解，但是它的基本含义是“只要你观察得足够仔细，总能发现点什么”，即使这种从训练数据上得到的结论不能推广到一般情况。

为了避免过拟合，可以采取交叉验证和简化模型的方式。图 13-3 展示了一种标准的情形（需要注意的是，实际工作中一般面对的是高维度空间，通常没有这样的图形可看）。

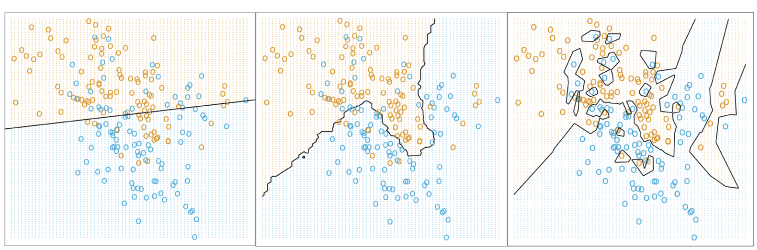


图 13-3：这幅经典的图片来摘自 Hastie 和 Tibshirani 合著的 *Elements of Statistical Learning*（《统计学习基础》，Springer-Verlag，参见 http://stanford.io/17szrYz），展示了同一份数据，对二值响应拟合线性回归模型时，采用 15 个最近邻和 1 个最近邻得到的不同结果（另见彩插图 13-3）

左图有点欠拟合，中间一幅刚好，右图过拟合。

将过拟合考虑在内，选择哪种模型差别很大，如图 13-4 所示。

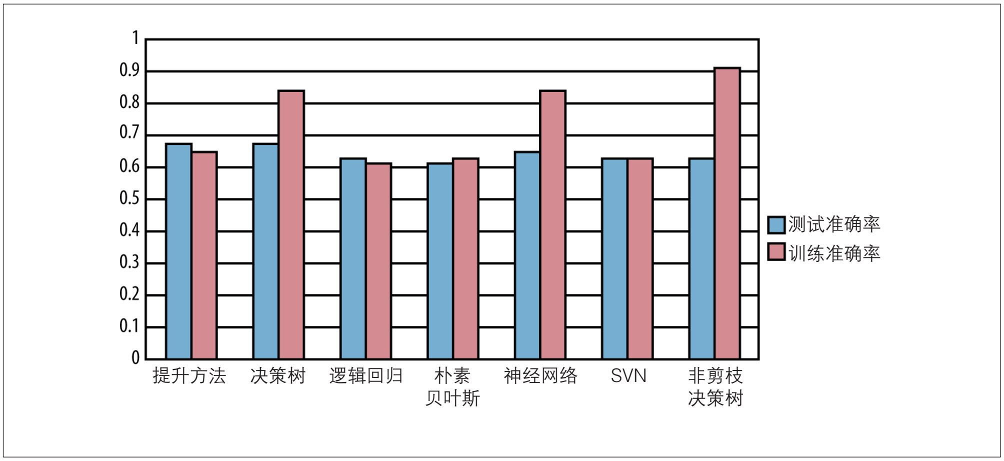


图 13-4：模型的差别（另见彩插图 13-4）

仔细看图 13-4，非剪枝决策树是“最过度拟合”的（“最过度拟合”是我们现发明的词），这是非剪枝决策树中广为人知的一个问题，因此，人们通常使用剪枝决策树。

13.6.1 准确度重要吗

在本书的讨论中，评价模型好坏的标准之一是准确度，尤其是对于那些二元分类问题。Claudia 认为准确度不是衡量模型的一个好的指标，使用准确度有什么问题？首先，它显然不适合用来评价回归模型；其次，对于那些大部分输出为 1 的二元分类模型也不适合，一个很傻的模型可能拥有很高的预测准确度，却不是一个好的模型（它预测所有的输出都是 1），一个好的模型却可能拥有较低的准确度。

13.6.2 概率的重要性，不是非0即1

没有人会根据二元输出做决策。你想知道的是罹患乳腺癌的概率，而不是一个是或不是的答案。概率包含了更多的信息。人们更看重概率。

那么 Claudia 认为评价一个模型好坏的标准是什么？她支持将排名和标定分开评价。为了评价排名，可以使用 ROC 曲线计算曲线下面积，通常是位于 0.5 和 1.0 之间的值。这是独立于标定的。图 13-5 展示了如何绘制 ROC 曲线。

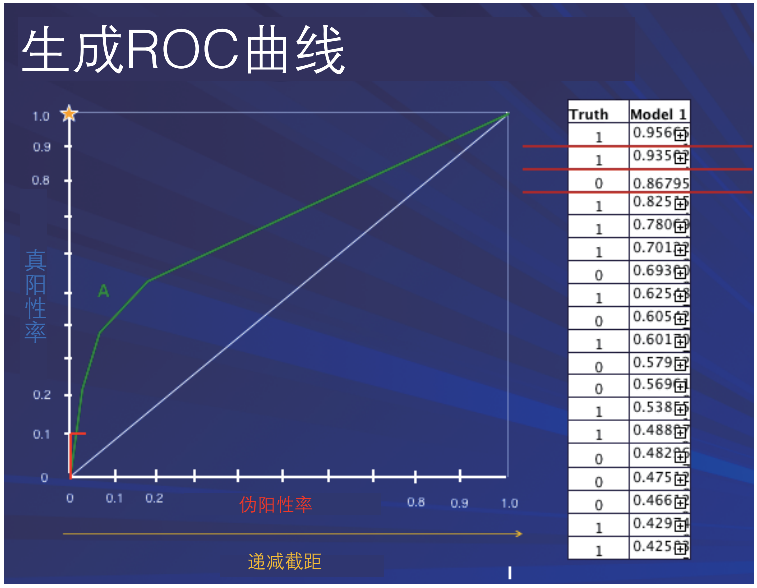


图 13-5：一个绘制 ROC 曲线的例子（另见彩插图 13-5）

评价排名时，有时还可以使用升力曲线，如图 13-6 所示。

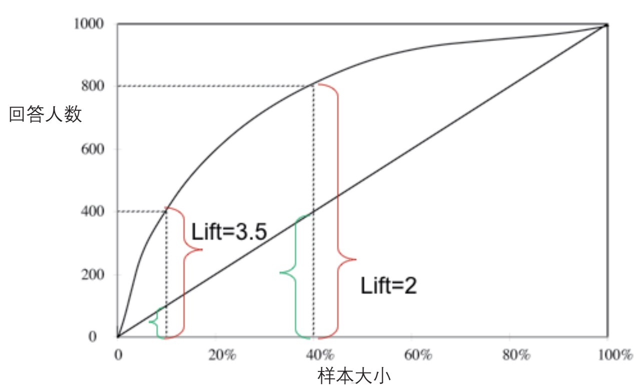


图 13-6：升力曲线（另见彩插图 13-6）

升力曲线的关键是升力是基于一个基线计算出来的。对于给定的一个点，比如 10%，想象一下，给 10% 的用户播放广告，对比随机选取 10% 的用户播放广告，哪种情况用户的点击量更多。升力为 3 表示点击量多 3 倍。

如何评价标定？概率准确吗？如果模型预测罹患癌症的概率为 0.57，怎么知道概率真是 0.57？我们无法直接衡量这个结果，只能将这些概率合并到几个分组中（比如 0.50~0.55），然后将它们和实际值进行对比。

图 13-7 展示了使用非剪枝决策树模型时的情形，图中蓝色的菱形对应相应的分组。

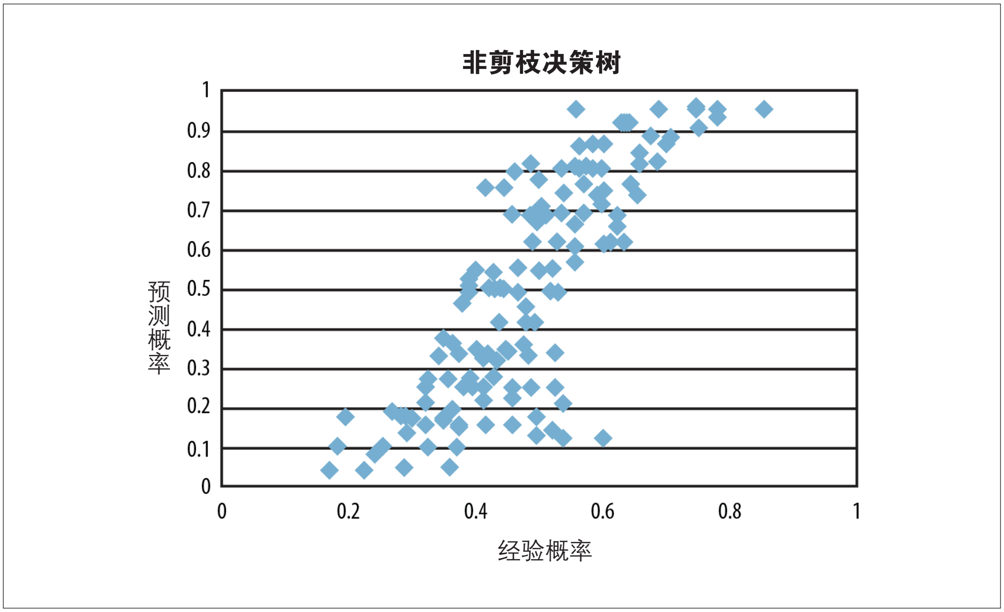


图 13-7：评价标定的方法之一是将模型预测得到的概率分组和经验概率分组绘制成散点图进行比较，这里我们使用的是非剪枝决策树模型

蓝色的菱形表示人群；*X* 轴是经验概率，表示经观察确定罹患癌症的人群；*Y* 轴表示利用非剪枝决策树模型预测出来的患病者概率的平均值。该图显示，一般情况下，决策树模型并不是一个评价标定的好标准。

一个好的模型中，分组应该分布在 $x = y$ 曲线附近，但是这张图显示预测概率明显高于实际概率。为什么决策树模型中会有这样的问题？

Claudia 解释道，这是由于决策树模型追求优化纯度的特性导致的，决策树里非 0 即 1。因此，相比现实情况，使用决策树模型得到的预测往往显得更极端。这是决策树的普遍问题：它们不适合评价标定。

而 Lotistic 回归就好很多，结果通常如图 13-8 所示。

蓝色的菱形依然表示人群。下图显示了逻辑回归模型在评价模型标定方面做得更好。

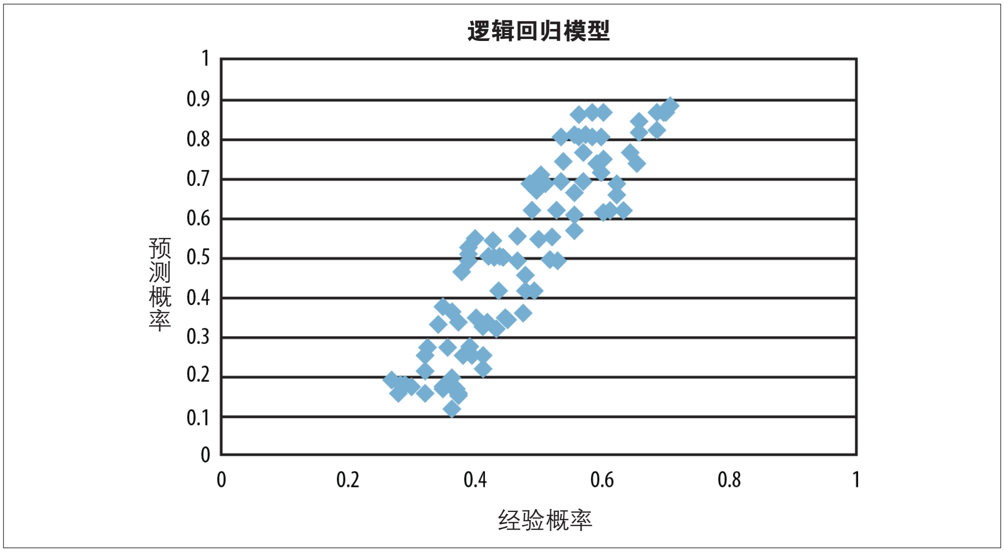


图 13-8：测试逻辑回归模型的标定

13.7 如何选择算法

这并不是一个容易回答的问题，特别是基于小数据集所做的测验往往会让你误入歧途。最佳算法会根据样本容量而发生变化。决策树通常表现不错，但前提是拥有足够多的数据。

一般情况下，需要根据数据集的大小和性质选择算法，选择评价模型的方法则要根据数据和你对模型的期望，你希望它在哪方面表现良好。如果数据服从正态分布，那么误差平方和就是最大似然估计的损失函数；如果要估计中位数，则应该使用绝对误差；如果要估计分位数，则应使加权绝对误差最小化。

我们曾经参加过一个比赛，预测电影在下一年的打分情况，我们假定打分情况服从泊松分布。这时，我们采用的评价方法就不包含让误差平方和最小化，而是包含了泊松分布中特有的属性，该属性依赖唯一的参数 λ。因此，有时候需要根据自己的情况，挑选那些适用的评价方式。

13.8 最后一个例子

让我们做个总结。

假设你要为慈善活动募捐。通过向邮件列表里的所有人群发邮件，你希望最后能募得 9000 美元资金。考虑到一般情况下只有 5% 的人愿意捐款，为了节省开支，你只给这些愿意捐款的人写信。请问，你如何知道哪些人愿意捐款？

如果选择裁剪过的决策树模型（这是一种标准做法），你的收益将会是 0，决策树里找不到一个大多数正向反馈的叶子节点。

如果选择神经网络，即使只给那些预期收益大于开支的人写信，也只能得到 7500 美元的收益。

让我们试着将这个问题分解如下。收到信的人通常会做两个决定：首先，他们要决定是否捐赠；其次，捐赠多少。我们可以使用如下公式为这两个决定分别进行建模：

$$E(\$ \mid person) = P(response = \text{`}yes\text{`} \mid person) \cdot E(\$ \mid response = \text{`}yes\text{`}, person)$$

要注意的是，必须确保第一个模型的准确性，因为你真正关心的是决定捐赠的人数，而不仅仅是捐赠数目。因此，对于第一部分，可以采用逻辑回归；对于第二部分，可以在同意捐赠的人里训练模型。

合起来，这种分解的模型能获得 15 000 美元的收入。通过分解的方式，模型可以很容易地捕捉到有用的信息。如果数据是无限的，一切都没有问题，你也不需要分解模型。但是现实并不完美，你需要结合实际进行工作。

此外，采用分解的方式也多次引入了误差，如果你有理由相信它们是相关的，那这可能会成为一个问题。

13.9 临别感言

Claudia 说，人类不是生来就懂数据的。数据处在我们的感官系统之外，只有极少数人对于数字有种天生的敏感。我们大多数人是命中注定只能理解语言的。

我们也不是生来就懂不确定性的：由于心存各种偏见，我们难以对不确定性做出正确理解，这些都已经被很好地记录下来。因此，从本质上来说，预测未来要比对已发生的事情进行分类难得多。

即使如此，我们仍竭尽所能，小心翼翼地产生数据，一丝不苟地理解问题，确保使用和真实环境接近的数据进行建模，确保朝我们期望的方向进行优化，不断学习，掌握哪种任务适用哪种算法。

第 14 章

数据工程：MapReduce、Pregel、Hadoop

本章由 David Crawshaw 和 Josh Wills 合作完成。Rachel 在谷歌工作期间，曾和他们一起共事，服务于 Google+ 项目的数据科学小组，但是他们俩并未一起工作过，Josh Wills 离开谷歌加入 Cloudera 后，David Crawshaw 接替他成为新的技术领头人。我们可以称他们为“数据工程师”，虽然这个称谓像“数据科学家”一样语焉不详（甚至很可能被过度解读），但可以这样说，他们是能够处理海量数据的软件工程师。根据第 2 章讲到的数据科学的工作流程，David 和 Josh 在谷歌的职责是收集数据（前后台日志），搭建海量数据管道以存储和整理数据，搭建用于分析、显示、A/B 测试，或者说进行数据科学研究的基础架构。

本章我们将看到来自谷歌工程师的关于 MapReduce 的一手信息，MapReduce 本身就是在谷歌开发的，随后涌现出了各种版本的开源实现。MapReduce 是一种算法和框架，用来处理渐在工业界流行起来的海量数据。本章的目的是揭开蒙在 MapReduce 上的神秘面纱。MapReduce 现在都成了时髦术语，很多招聘数据科学家的职位描述上都要求应聘者“必须掌握 Hadoop”（Hadoop 是 MapReduce 的一个开源实现）。我们怀疑这些招聘信息是人力资源部的人写的，他们根本就不知道 MapReduce 是用来干什么的，他们也不知道不是所有的数据科学问题都要用到 MapReduce。但是既然 MapReduce 已经成为一个数据科学的术语，我们想讲清楚它到底是什么、它从哪儿来的。你应该了解 MapReduce，但是未必会用到它，这完全取决于你的工作内容。

数据科学家需要知道 MapReduce 吗？

来玩一个好玩的游戏：去参加一个数据科学的会议，数数“MapReduce”被提及了多少次，然后向他们提问什么是 MapReduce，看看有多少人能解释清楚。我们怀疑，即使这些参会的专家在工作中已花费数不清的时间来应用 MapReduce，但真正能解释清楚的人也不会太多。在谷歌工作期间，Rachel 使用 Sawzall 语言写了一些代码来处理和整理数据，以便进行分析和搭建原型，Sawzall 语言将 MapReduce 框架内建于它的逻辑结构之中。Cathy 也曾在 Intent Media 担任数据科学家期间使用过 Pig——一个 Sawzall 的开源版本，她在 Mortar Data 框架中同时使用了 Pig 和 Python。我们确实间接地用过 MapReduce，我们也确实理解它，但是肯定没有这些亲自实现 MapReduce 的家伙理解得透彻。

讨论 MapReduce 的另一个原因是，它说明了在大数据情景下，解决工程和基础架构问题时所使用的一类算法。这是我们在第 3 章中提出的第三类算法（其他两类是机器学习算法和优化算法）。为了对比，我们还介绍了另一种数据工程算法和框架——Pregel（Pregel 也来自谷歌，而且已经开源），它对工程师的算法知识要求不高，支持大规模图的计算。

14.1 关于David Crawshaw

David Crawshaw 是谷歌的一名软件工程师，他曾经使用了一段错误的 Shell 脚本误删了 10PB 数据，幸运的是，他有备份。David 学的是数学，曾和 Rachel 一起在加利福尼亚为 Google+ 项目工作，现在他的工作是为更好地理解搜索搭建基础架构，最近他从旧金山搬到了纽约。

David 为我们介绍了 MapReduce，以及如何处理海量数据。在深入这些内容之前，让我们先做一个思维实验。

14.2 思维实验

我们应如何看待开放查阅病历记录和保护隐私之间的关系？

一方面，将病历记录开放查阅会带来严重的隐私侵犯问题——我们不想让随便一个人都能得到用户的医疗历史。另一方面，有时候获取这些信息则可以挽救患者生命。

据估计，在一个相当小的镇上，每周都有一到两个患者死于信息匮乏，因为其医疗记录不能及时在医院急诊室和附近的精神卫生诊所流通。换句话说，如果这些记录可以非常方便地匹配起来，那么将会有更多的生命得到挽救。另一方面，如果匹配这些记录很容易，一些保密的信息则有可能泄漏。当然，我们很难知道具体有多少生命因这个原因而危在旦夕，但绝不是个小数。

这就自然引出一系列问题：医疗记录里有多少信息属于隐私，是需要保护的；谁有权力访问你的医疗记录；在什么条件下才能访问。

我们假设尊重隐私一般情况下是件好事。比如，无神论者在某些地方是要被判处死刑的，在这种情况下，最好保护这些隐私。但有时候隐私也会带来死亡，就像我们前面讲到的急诊室里因缺乏这些信息导致患者死亡的案例。

让我们再来看看其他例子，执法机关时常要面对一些时违法犯罪行为。比如说，执法人员手上掌握着大量的汽车牌照数据，如果被别有用心的人有权限访问，他们有可能会滥用这些信息。在这种情况下，就不是技术的问题，而是人的问题。

这同时也是一个哲学问题：在何种程度上我们可以代替他人做决定？

这里还有一个主观意愿的问题，拥有更多的医疗数据我们或许可以更快地治愈癌症，但是如果患者不愿公开病史，我们也不能因此拒绝治疗他们。

最后，对个人来说，这也是一个安全问题。通常情况下，人们不在意别人拥有这些数据，人们在意的是这些数据是否会给他们造成伤害，或者将数据和他们个体关联起来。

再回到技术问题，搜集数据要做到完全匿名是非常困难的。Alexandre de Montjoye 等人在《自然》杂志上发表了一篇文章“Unique in the Crowd: the privacy bounds of human mobility”（“群体中的独立性：移动化趋势中的人类隐私权界限”），研究发现，对于一份包含 150 万欧洲用户手机号码的数据，仅仅四位数就可以区分 95% 的人。

最近我们发现人们在竭力反对 NSA 收集美国公民数据的行为（更别提搜集非美国公民的数据了）。事实上，在本书就要付梓的时候，Edward Snowden 爆出了“棱镜门”。这引发了一场广泛且激烈的大辩论，各方激辩政府权力与公民隐私之间的边界。想想现在有多少个人信息通过信息数据仓库和 Acxiom 这样的经纪公司在网上售卖，这些信息不光包含市场信息，还有保险、职业、贷款信息。我们或许也应该就这些问题和企业展开类似的对话。

14.3 MapReduce

让我们先来看看 David 是如何站在工程师角度来看待大数据的。

大数据更多意义上只是一个时髦术语，但是它是有用的。David 试图用如下的话定义大数据：

当你处理数据时，如果一个计算单元容纳不下这些数据，则这就是大数据。这是一个随时间而演进的定义，按照这个定义，大数据已经存在很长时间了。早在电脑发明之前，美国国家税务局就在征税，根据刚才的定义，这些税收数据也无法放在一个（不存在的）计算单元里，因此也可以叫作大数据。

现在，大数据指不能使用一台计算机处理的数据，即使如此，大数据的数量也在快速地增长。计算机在过去四十年中呈指数级增长，现在至少还可以以这个速度持续增长十年（十年前人们就这样说）。

既然这样，大数据会消失吗？我们可以忽略它吗？

David 认为我们不能这样想。因为虽然计算机的处理能力也在保持指数级增长，但是这些计算机同时在以同样的处理能力产生数据。新的数据也在以指数级的方式增长。因此有两条指数级增长的曲线，短期内还看不到相交的可能。

让我们用一个实例来看看情况会变得多么复杂。

14.4 单词频率问题

假设要从下述列表中找出出现频率最高的词：red, green, bird, blue, green, red, red。

最容易的方式当然是直接通过肉眼观察，但是假如这个列表变长了，变成包含 10 000 个、100 000 个、1 000 000 000 个单词的列表，这个时候怎么办？

最简单的方式是写程序将所有单词列入其中，计算每一个单词出现的次数。图 14-1 展示了使用 Go 语言（http://golang.org/）编写的代码片段，Go 是 David 喜欢的语言，他在谷歌参与了该语言的设计与实现（你实现过哪种语言吗）。

```
func count (words []string) {
        counts:- map[string]int{}
        for _, word:= range words{
                counts[word] +=1
        }
        printSorted(counts)
}
```

图 14-1：使用 Go 语言实现的代码片段

计数和排序是很快的，因此这个算法有能力处理包含 1 亿数量级单词的列表。问题的瓶颈在内存——试想一下，列表中的单词需要两次被装载进内存中：装载列表时一次，为每个单词计数时一次。

可以对程序做一些改进，不必一次装载整个列表，将列表存放在磁盘里，使用管道代替列表，在需要时以流的方式读入。管道像是一种流：读取排在最前面的 100 个单元，处理完后再读取后 100 个。

但是这仍然没能解决潜在的问题，如果列表很长，且列表中的每个单词都不同，内存还是放不下，程序依然会崩溃。另一方面，这段程序可能在绝大多数情况下都可正常工作，因为大多数情况下总会有重复的单词出现。写程序就是这样一种混乱的游戏。

先别急，现在的计算机不都是多核的吗？让我们把它们全用上！这时，带宽又成了问题，让我们再把输入进行压缩。不能说这些方法不管用，但是它们增加了复杂性，还有比这些更好的方法，但是复杂性更高。存储散列值的定长的堆表现就不错。堆是一种数据结构，有点类似半排序的集合，可以扔掉那些极小的单元，以避免将所有东西都放在内存里。这种方法并非每次都行得通，但大多数情况下是奏效的。

还能跟上吗？

你不必清楚这里的细节，我们只是想让你知道为什么需要 MapReduce。

现在，使用一台计算机可以处理 10 兆数量级的单词。假设现在有 10 台电脑，那么可以处理 100 兆数量级的单词。每台电脑处理 1/10 的单词，然后将结果汇总到一台“主控”电脑。主控电脑负责将结果汇总，求出出现频率最高的单词。

如果我们学过网络编程，也可以使用存储散列值的堆来解决这个问题。

假设现在有 100 台电脑，那么就可以处理 1000 兆数量级的单词。不过问题又来了，由于带宽有限，将每台电脑的处理结果发送到主控电脑时又行不通了。这时需要一个树形拓扑结构，以 10 台电脑为一组，将结果发送给一个局部的主控电脑，然后这 10 台局部的主控电脑再发送到最顶级的主控电脑，问题可能就解决了。

但是，这样的方法扩展到 1000 台电脑时是否同样有效？答案是否定的，这种方式行不通。这么多电脑，总有一两个会坏掉，假设用 X 表示一台电脑是否工作正常，$X = 0$ 表示工作正常，$X = 1$ 表示不正常，那么所有电脑工作正常的概率为：

$$P(X = 0) = 1 - \epsilon$$

这同时意味着，这 1000 台电脑中没有任何一台工作不正常的概率为：

$$(1 - \epsilon)^{1000}$$

即使 ϵ 足够小，这个概率也是非常小的。假设电脑出错的概率 $\epsilon = 0.001$，那么 1000 台电脑全部工作正常的概率为 0.37，还不到一半。这种架构显然不够坚固。

那么我们应该怎么做？

想要弄明白这个问题，先得说说分布式系统的容错功能。这通常需要对输入进行备份（默认将输入复制成三份），将每个备份交给不同的电脑处理，如果一份数据处理时损坏了，另外一个备份依然能保证数据完整。还可以将校验码嵌入数据中，这样数据本身就具有了校验错误的能力，我们就可以用一台（或者多台）电脑来实现自动化管理。

一般来说，需要开发一个可以探测到错误、并能自动恢复的系统。为了提高效率，当某些机器执行完任务后，可以重复执行其他任务，当然，免不了还要检测错误。

问：等等，我记得我们是在说计数。现在，我觉得我们又惹上了另一个麻烦。
答：事情总是这样的，论证容错性的效率并非易事，所有的事情都很复杂。要注意的是，效率和正确性同等重要，你的薪水可比 1000 台电脑值钱。

看起来像是这样：

- 对付 10 台电脑很容易；
- 对付 100 台电脑有点难；
- 对付 1000 台电脑简直是不可能完成的任务。

没有一点办法!

起码，在 8 年前是这样。现在，David 在谷歌动辄就使用 10 000 台电脑组成的集群。

初涉MapReduce

2004 年，Jeff 和 Sanjay 联合发表了一篇论文“MapReduce: Simplified Data Processing on Large Clusters”（“MapReduce：大集群上简化的数据处理”），同时还有另外一篇论文“The Google File System”（“谷歌文件系统”，参见 http://research.google.com/archive/gfs.html）描述了 MapReduce 架构的底层文件系统。

MapReduce 是个平台，在这个平台上，程序员不用再考虑处理容错性，平台本身就为我们提供了这种服务。它也是一个函数库，用它可以做很多过去不敢想的事。现在，使用 MapReduce，在 1000 台电脑上编程反而比过去在 100 台电脑上还简单。

使用 MapReduce 需要写两个函数：一个 mapper 函数，一个 reducer 函数。这两个函数会在分布存储数据的各个电脑上运行，只要你将算法放进这种 map/reduce 的框架，系统就自动具备了容错的功能。

mapper 函数将每个数据点作为输入，按顺序输出形如 (键 , 值) 这样的二元组列表，然后框架会使用定向冒泡排序对输出排序，如果键值相同，会将两个二元组合并，将值堆叠起来。这些堆最终会被送往执行 reducer 函数的机器。reducer 函数使用聚合函数对值进行聚合，得到新的值，输出新的 (键 , 新值) 列表。

来看看如何使用这种方式解决上面提到的单词计数算法。以每个单词为键，值设为 1，则有如下的二元组：

```
red ---> ("red", 1)
blue ---> ("blue", 1)
red ---> ("red", 1)
```

然后对其排序，得到一堆 ("red", 1)，记为 ("red", 1, 1)，然后被送往 reducer 函数，reducer

函数将所有的 1 相加，最终结果为：("red", 2), ("blue", 1)。

这里的关键是：一个 reducer 处理给定键的所有值。

如果数据变得越来越多怎么办？只要增加 mapper 和 reducer 的数量就行了。换句话说，使用更多的电脑。MapReduce 抹平了在多台电脑上工作时的复杂性。它是如此优雅，以致人们在不需要的时候也用它（然而在谷歌，假设数据每晚呈 100 倍地增长，这种假设并不算疯狂）。像所有的工具一样，MapReduce 被滥用了。

计数本身是一个简单的函数，现在被拆分成了两个函数。一般情况下，如何将一个算法拆解成一些列的 MapReduce 步骤并不是那么直观的。

对于上面提到的单词计数问题，单词必须服从统一分布。如果所有的单词都一样，那么在排序阶段都会汇入同一台机器，这是个大问题。谷歌使用另外一种数据结构“散列存储桶堆”解决了这个问题，在每个 MapReduce 迭代的 mapper 函数中使用该数据结构。谷歌给它起了个名字，叫“CountSketch”，专门用来处理奇异数据集。

在谷歌，有一个监控器实时监控 MapReduce 作业的状态，绘制成一个条状图，每一条和机器上的一个任务对应。如果每个 mapper 函数运行正常，条状图看起来就像一条直线。但是通常情况下，这种现象并不多见，由于数据并不服从统一分布，一个键对应太多的值，在 reduce 阶段，所有的事看起来都不对了。

后台需要运行数据准备和写文件，这通常要花很长时间，因此最好在一个迭代里将所有事做完。这里假设分布式文件系统已经就位，我们需要使用 MapReduce 将数据写入分布式文件系统——上了 MapReduce 的船就下不来了。

当来到优化阶段，为一点细小的性能提升，比如在处理 PB 级的数据时将性能提升几微妙，可能要绞尽脑汁。提升几微妙，这通常是物理学家才会如此关心的事。这部分优化是用 C++ 实现的，这是一段高度优化的代码，我们竭尽所能将系统的性能发挥到极致。

14.5　其他MapReduce案例

单词计数只是 MapReduce 最基本的案例之一，为了对 MapReduce 有一个更全面的认识，让我们看看其他一些例子。能用 MapReduce 解决的问题有一个重要特征：数据可以被分布到多台计算机上，且算法可以在每台计算机上独立处理数据，即每台计算机之间是相互隔离的，它们不需要知道其他机器在处理什么。

下面是能使用 MapReduce 的另外一个案例。假设现在有记录用户访问某网站行为的数以万计的数据，对于每个用户，有如下形式的一行记录：`{user_id, IP_address, zip code, ad_they_saw, did_they_click}`。假设你想知道不同行政区域（根据不同的邮政编码）看到广告的用户数量，和看到广告并且至少点击一次的用户数量，而且不能将一个用户重复计算。

怎么使用 MapReduce 处理上述问题？在执行 MapReduce 作业时，以邮政编码作为键，假设一个用户所在区域的邮政编码为 90210，如果他看见广告并且点击了，则有 `(90210,{1,1})`，如果他看见了但是没点击，记为 `(90210,{0,1})`。

这能带给你什么呢？在 reducer 阶段，会以邮政编码为组，计算用户看到广告和看到广告并且点击的次数，产生形如 `(90210,{700,15530})` 这样的输出。但是，这不是题目要求的，题目要求用户不能重复计算。这需要两个 MapReduce。

第一个 MapReduce 使用 `{zip_code,user}` 作键，`{clicks, impressions}` 作值，比如 `({90210,user_5321},{0,1})` 或者 `{90210, user_5321} <- {1,1}`，reducer 会根据每个用户、每个邮政编码生成一个包含点击次数和看见次数的表格，形如 `{user, zipcode, number_clicks, number_impressions}`。

然后计算来自每个行政区域的不同用户的数量，他们至少点击过一次广告。这时需要第二个 MapReduce，以邮政编码作键，`{1, ifelse(clicks>0)}` 作值。

这是使用 MapReduce 计数的另一个演示，但是 MapReduce 能做点更复杂的事吗？比如实现一个统计模型，线性回归什么的。这可能吗？

答案是肯定的。2006 年发表的一篇论文阐述了如何使用 MapReduce 实现各种类型的机器学习算法（http://machinelearning.wustl.edu/mlpapers/paper_files/NIPS2006_725.pdf）。有些算法在计算各种统计量和变化率时需要先计算期望值和求和，此时就可以使用该论文描述的方法。因为这些计算可以批量处理，可以表示为不同数据点上的和。

MapReduce不能做什么

有时候，了解一件事是什么，可以帮助了解它不是什么。那么，什么是 MapReduce 不能做的呢？不能做的事太多了，比如给我们发个消息。认为所有和数据相关的问题都能使用 MapReduce 的方式解决，这也是可以谅解的。

但是 MapReduce 并不适合迭代式的算法——计算时将上一次的输出作为下一次计算的输入，很多机器学习算法都如此，它们在计算最速下降收敛方法是都用这样的方式。如果你非要坚持使用 MapReduce，这仍然是可能的，但这需要在引擎里四处修改。有一些新的方法更适合处理这种情况，比如 Spark，它执行起来效率更高。

14.6 Pregel

为了和 MapReduce 对比，我们再介绍另一种处理大数据的算法：Pregel。该算法同样出自谷歌，它是一种基于图的计算方法，这里可以将数据想象成一种图状或网状的数据结构，连通的节点之间可以互相交换信息。同时还有聚合节点，可以获取所有节点上的信息，然

后在其上进行求和或计算平均数之类的运算。

该算法的基础是运行于节点之间的一些超级步骤，它将信息从一个节点送往另一个节点，从一般节点送往聚合节点。该算法的论文在网上可以找到（http://dl.acm.org/citation.cfm?id=1807184），同时该算法还有一个叫 Giraph 的开源实现（http://giraph.apache.org/）。

14.7 关于Josh Wills

Josh Wills 是 Cloudera 的数据科学主管，带领工程师为来自各个行业的客户提供基于 Hadoop 的解决方案，稍后会介绍更多关于 Cloudera 和 Hadoop 的内容。在加入 Cloudera 之前，他为谷歌工作，在那里开发广告竞拍系统，后来领导开发了 Google+ 项目的数据分析基础架构。他在杜克大学取得了数学学士学位，在奥斯汀的田纳西大学取得了运筹学硕士学位。

Josh Wills 因对数据科学发表的一些精辟格言而闻名，比如，他曾说“我让数据发光发热”，本书开始也引用了他对数据科学家的定义“数据科学家 (名词)，是软件工程师里最懂统计的，统计学家里最会编程的”，还有“我是阿甘，我有一柄牙刷，我有很多的数据，我的工作就是对着它没日没夜的刷”。

Josh Wills 以一个思维实验引入了他的主题。

14.8 思维实验

如何建造一架人力飞机？你会怎么做？你应该如何组建一个团队？

也许你会搞个有奖竞赛（X prize，参见 http://www.xprize.org/）。有人就是这样干的，他们在 1950 年悬赏 5 万美元。十年后，才有人拿到了那笔奖金。获胜者的故事值得我们在这里讲述，因为它说明了人们有时候致力于解决的问题本身就是错误的。

最初的几个团队花了几年时间做计划，然后飞机飞上天几秒钟就落下来摔得粉碎。而赢得比赛的那个团队则换了个思路：如何建造一架出事后在 4 小时内就能恢复的飞机？经过几轮快速的原型迭代，他们成功了，只用了 6 个月。

14.9 给数据科学家的话

通过对数据科学家日常工作的观察，Josh 发现 90% 的时间都被用来清理和准备数据。在解决问题和获得洞见之间，数据科学家往往选择前者。更确切地说，从一个问题出发，确保它有值得优化的地方，然后并行化处理所有事情。

天资聪颖当然是件好事，但是能够快速学习更好，立即着手实验，立即从中学习到有用的东西。

14.9.1 数据丰富和数据匮乏

大多数人偏向保守，他们选择丢弃那些看似无用的东西，习惯使用有限的数据思考问题。Josh 选择保留一切。他是可重复性研究的粉丝，所以他希望能够重现分析过程中的任何阶段。这样做有两个原因。首先，如果犯错，他不用把一切推倒重来。其次，当有新的数据时，很容易集成到当前工作流程中来。

14.9.2 设计模型

模型不免最终沦为疯狂的鲁布·戈德堡机械（Rube Goldberg machine，是一种设计得过度复杂的机械组合，以迂回曲折的方法去完成一些其实是非常简单的工作，例如倒一杯茶，或打一只蛋等），一堆模型的大杂烩。这并不总是件坏事，只要模型能工作，就没有问题。即使以一个简单的模型开始，最终还是会添添补补，变得复杂起来。这种事一再发生，没办法，这就是设计模型的本质。

认清分歧

为模型做优化和为业务做优化是不同的。

比如，Facebook 的朋友推荐系统并不用来帮助你甄选那些真正志趣相投的朋友，它是为了让你在 Facebook 网站上花费尽可能多的时间。你想想，推荐给你的朋友是不是都是些迷人的异性？

其他领域又是什么情况呢？在医疗机构，他们研究某种药物的疗效而不是患者的健康，他们通常关心手术是否成功，而不是患者的幸福。

14.10 算算Hadoop的经济账

让我们回过头来再看看 MapReduce 和 Hadoop。2001 年，Josh 刚从学校毕业，当时存储文件有两种选择：数据库和存储文件管理器，Josh 从模式、处理能力、可靠性和花销四个维度对二者进行了对比。

表14-1：2001年时可选的文件存储类型

	数据库	存储文件管理器
模式	结构化的	非结构化的
处理能力	强大的数据处理能力	没有数据处理能力
可靠性	可靠	非常可靠
花销	存储大规模数据时花费高	存储大规模数据时花费高

现在产生的数据越来越多，其中大部分来自网页。这自然引出了衡量投入产出比的经济学指标：字节价值。从一字节的数据里我们可以获取多少价值？存储它又要花费多少钱？如

果我们求两者的比值，则希望这个比值大于 1，否则不如丢弃这些数据。

当然，这不是故事的全部。有一个有关大数据的经济学定律：任何单条记录都不是特别有用的，但是拥有所有的记录则价值连城。比如，对于网页索引、推荐系统、传感器数据、在线广告等系统，如果拥有现存的所有数据是特别有用的，但单独的每个数据点其实是没有什么价值的。

14.10.1 Hadoop简介

在谷歌还没有像今天这样有钱时，它们的硬件简直差劲极了。为了处理数据，他们想出了一个主意，将数据拷贝到多台服务器上。一开始，他们手动拷贝，后来变成自动化的，这个自动化的过程后来发展成全局文件系统 GFS。

Hadoop 是谷歌的 GFS 和 MapReduce 的开源实现（读者可以在其他地方了解到关于 Haddop 起源的故事，这里我们只给出两点提示：有一个叫 Nutch 的开源项目，雅虎公司也参与其中），它的核心分成两部分。第一部分是分布式文件系统（HDFS），它是基于谷歌的文件系统。数据存储在大文件里，文件块的大小通常为 64 ～ 256 MB。这些文件块被复制到集群中的多个节点上。如果一个节点宕机，主控节点会得到通知。第二部分是 MapReduce，David Crawshaw 已经在前面讲过了，这里不再赘述。

Hadoop 是用 Java 实现的，谷歌的那一摊子是用 C++ 实现的。使用 Hadoop 提供的 Java API 编写 MapReduce 程序并不会令人感到愉快，有时候需要写大量的 MapReduce。但是，如果使用 Hadoop 流，则可以使用 Python、R 或其他高级编程语言。这使得编写并行任务程序变得简单方便。

14.10.2 Cloudera

Cloudera 是由 Doug Cutting 和 Jeff Hammerbacher 共同创建的，前一位是 Hadoop 的创始人之一，后一位我们在第 1 章提起过，在 Facebook 工作期间，他率先提出了“数据科学家”这个职位，并为 Facebook 组建了数据科学团队。

Cloudera 之于 Hadoop 就像 Red Hat 之于 Linux，同样都是围绕一个开源项目创立了一家公司。Hadoop 是由 Apache 软件基金会赞助的，代码是免费的，但是 Cloudera 将所有东西打包，并且免费提供各个版本，它靠为客户提供技术支持和服务赚钱，赚回来的钱反过来又可以反哺项目，使 Hadoop 得到更好的发展。

Apache Hive（http://hive.apache.org/）是建造于 Hadoop 之上的一个数据仓库系统，它使用类似 SQL 的查询语言（包括某些特定的 MapReduce 扩展），实现了常见的合并和聚合操作。对于精通数据库和熟悉这些操作的人来说，这是再好不过了。

14.11 Josh的工作流程

来看看 Josh 是如何使用 MapReduce 打造数据管道的，他将一条记录视为数据分析的基本单元。我们不止一次的提及“打上时间戳的活动数据”，你可以将每一条视作一项记录，或者像前面讨论过的交易记录，比如欺诈检测、信用卡交易。一个典型的工作流程如下：

(1) 使用 Hive（一种运行在 Hadoop 上的类 SQL 语言）将你对某实体的了解情况创建为记录（比如一个人）——这里会使用到大量的 MapReduce）；
(2) 编写 Python 脚本反复处理这些记录（速度要快，迭代要快，同时还会用到 MapReduce）；
(3) 当有新数据时，及时更新记录。

要注意第 (2) 步的脚本通常只是 map 作业，它让并行化变得简单。

Josh 更喜欢标准的数据格式：巨大的文本占据了空间，而 Thrift（http://en.wikipedia.org/wiki/Apache_Thrift）、Avro（http://en.wikipedia.org/wiki/Apache_Avro）和协议缓存（http://en.wikipedia.org/wiki/Protobuf）是一种更紧凑的二进制数据结构。Josh 还鼓励大家使用用来存储代码和元数据的 Github（https://github.com/），他不使用 Git 存储大文件。

14.12 如何开始使用Hadoop

如果你所在公司拥有 Hadoop 集群，很有可能你和 Hadoop 的第一次亲密接触是通过 Apache Hive，它在 HDFS 和 MapReduce 之上，提供了一种类似 SQL 风格的抽象层。你的第一个 MapReduce 作业很可能是通过对记录用户行为的日志进行分析，来更好地了解客户如何使用公司的产品。

如果你要使用 Hadoop 和 MapReduce 编写自己的分析应用，有很多途径可以上手。使用 Apache Mahout 搭建一个推荐引擎就是一个很好的选择。Apache Mahout 包含了一些机器学习的程序库和命令行工具，可以和 Hadoop 一起工作。它有一个叫作 Taste 的协同过滤引擎，使用它，给定一个 CSV 文件，包含用户 ID、物品 ID 和一个可选的，表征用户和物品联系紧密程度的权重信息，就可以创建一个推荐引擎。Taste 使用的推荐算法和 Netflix、亚马逊使用的推荐算法都是一样的。

第 15 章

听听学生们怎么说

“每一个算法都是一篇文章。”

—— Emily Bell，哥伦比亚大学新闻学院
数字新闻学塔尔中心主任

我们邀请了最早一批选修了数据导论课程的学生来写作本章，他们分享了对本课的想法和自己的学习经历。参与编写本章内容的同学有：Alexandra Boghosian、Jed Dougherty、Eurry Kim、Albert Lee、Adam Obeng 和 Kaz Sakamoto。

15.1 重在过程

开始学习数据科学时，你别无选择，只能从最前沿的地方开始，因为数据科学本身就处在科技发展的最前沿。

物理学的入门课程通常从易到难，先介绍经典力学、电磁学，然后过渡到近代物理，比如狭义相对论之类。但是这种循序渐进的教学方式并未揭示物理学原理的发现过程，比如，牛顿究竟是如何发明了微积分？没人教给我们这个发现的过程。牛顿是如何做到的？我们不知道他使用了什么工具，他读了什么书。他记笔记吗？他有没有尝试复现别人的证明？他集中精力解决的问题是否来源于他上次写的某篇文章？到底是什么促使他去想“我必须有极限才能解决这个问题”？牛顿演算时需要打草稿吗？还是说很多想法在他心中早已成型，看见苹果落地时碰巧都涌现出来了？这些东西是教不来的，但又是我们必须学会的，那些初出茅庐的科学家必须学会这个过程。

Rachel 在数据科学导论的第一堂课上就严肃告诫我们：无论在学术界还是工业界，数据科

学尚在定义之中。在接下来的课程里，我们遇见了实实在在的问题，并且看到老师们是如何在这些问题中选择他们想要研究的课题。实际上，每周的课程都覆盖了数据科学家需要用到的工具和技术，只是每节课都有它自己的风格和背景，解决问题的方式也不一样。几乎在每一节课上，老师们都会说："我不知道你们之前都学了些什么，但是……"课程是离散的，我们要负责将这些残章断片拼成一个关于数据科学的完整介绍。我们要从课程里发掘出对自己有用的部分，就像数据科学家们在持续不断地建造他们自己的领域。

这并不是说 Rachel 有意要将我们蒙在鼓里。在课程的第一天，她就提出了一个数据科学家的定义，她说，数据科学家就是兼具如下能力的人：数学、统计学、计算机科学、机器学习、可视化、沟通的技巧和行业知识。随后我们马上发现，就像我们理解的贝叶斯定理一样，这只是些先验知识。同学们和讲师们都根据这个定义，对自己做了评估，为数据科学界提供了多样性的写照，并且这份评估也作为参考，贯穿课程始终。担任该课的讲师们来自学术界、金融业、科技公司和初创公司，他们有的在研究生期间就中途退学，有的则是 Kaggle 竞赛的获胜者，还有数字艺术家。每个人都提供了更进一步的似然比。课程本身就在对数据科学进行迭代式的定义。

但是我们并不是每周都在课堂上听他们谈论自己的工作。通过完成大量很难的作业，我们学会了数据科学研究所使用的工具。有时候，作业要求实现课堂上讨论的技术和概念，有时候我们则需要自己去发现和探索还不知道的技术。

另外，我们面对的是混乱的真实数据。我们经常要去解决工业界内的真实问题，并且需要最终形成一份清晰和深刻的报告。这份报告，即使拿给业界的专家看，我们也会感到自豪。最重要的，为了完成作业，我们经常要跨出自己的舒适区。在这里，强调了数据科学的社会性，作业和项目都需要分成小组，大家合作完成。Rachel 还常常带领我们和当日的讲师，穿过街道去酒吧坐坐[1]。就这样，在整个学期里，我们大家一起工作，一起喝酒，互相学习，共同学习从事数据科学工作需要的技能。

15.2 不再简单

真正的挑战来了，第二个作业（4.7 节）的第三小节，要求从《纽约时报》的官方网站上下载 2000 篇文章，并且通过这些文章训练出一个朴素贝叶斯分类器，按文章出现的不同版面对其进行排序。但是那个网站一次只允许下载 20 篇文章！抓取这些文章只是万里长征的第一步，我们还要亲自编写代码实现贝叶斯公式，唯一能帮我们的也就是老师给出的几个公式。很多编程语言都有现成的实现，倒不是说这些实现对我们没有帮助，我们的实现有一些特别的需求，我们要求正则化的超参数是可调的，作业还要求将文章分为 5 类，而不是两类。我们听说朴素贝叶斯并不是那么"朴素"，也不是很"贝叶斯"，它的实现并

注 1：Rachle 注释说"这是一门研究生课程，我确保学生们都到了可以喝酒的年龄，而且为不喝酒的同学我们也提供了非酒精类饮品。"

没有用到贝叶斯模型。结果正是如此，朴素贝叶斯不简单。然而，我们中的有些人还是坚持下来了，他们在教室里熬了 40 个小时去打磨一段大概 300 多行、令人反胃的 R 代码，最终使模型的预测准确度达到了 90%，别提有多高兴了！我们立刻对这件事上瘾了，当然，也可能是我们不忍心丢掉沉没成本——那些在这个作业上花的精力和时间。总之，我们是上瘾了。图 15-1 是一位同学的答案。

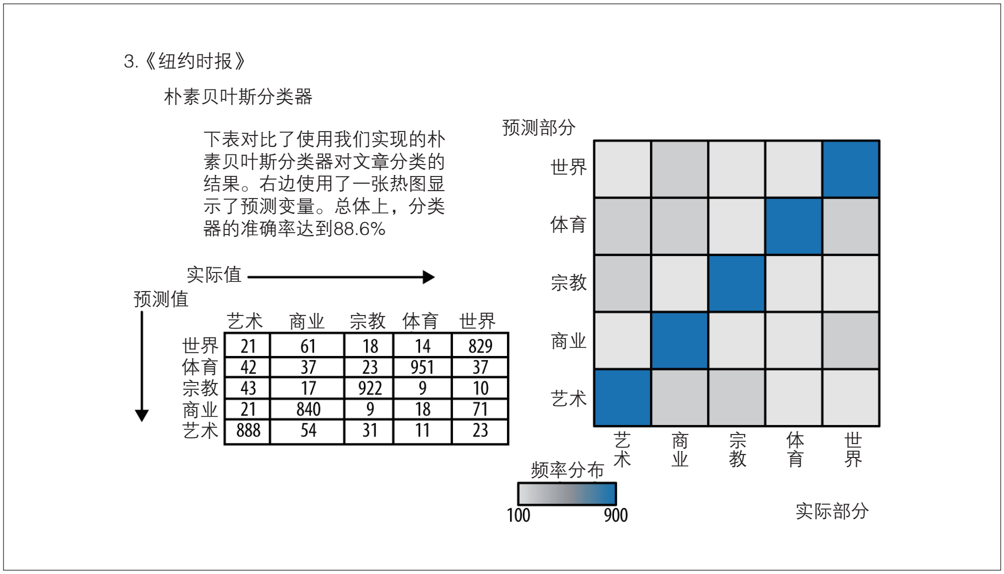


图 15-1：一位同学的部分答案（另见彩插图 15-1）

参加 Kaggle 竞赛是最后的大作业的一部分，它给了我们另辟蹊径的机会。作业是在学生们中间进行一场竞赛，设计一个论文评分的算法。一般的作业大多模拟工业界的数据科学家们的工作内容，而 Kaggle 竞赛是数据科学界的一个一决雌雄的比赛，它鼓励我们将课堂上学到的所有关于数据科学的最佳方法都应用上，同时又使得我们可以亲历经典的数据科学活动。本书的作者之一的方案引入了如下特征变量：拼写错误的个数、Dale-Chall 单词列表（这是一个四年级学生应该掌握的词汇表）、论文中最常出现的签 50 个单词的 TF-IDF 向量、论文单词数的四次方根。别问我为什么，方案使用了随机森林模型和 Gradient boosting 算法，并且实现了随机超参数优化，在亚马逊提供的 EC2 平台上运行了几千个小时，训练了 5 万个模型，它很管用。

15.3　援助之手

在课程的前半程，我们遇见了客座讲师 Jake Hofman。你还记得第一次看见魔术时的情景吗？没错，Jake 的课就像在你眼皮底下表演魔术。仅仅用了一些简单的 Unix 系统命令和数据，他就在我们面前活生生写出了一个朴素贝叶斯垃圾邮件分类器。在黑板上写了一串

数学公式后，他现场下载了安然公司的 email 数据，然后通过高超的命令行技巧分析了这些数据。

在每周一次的课堂上，都有超级棒的讲师们来给我们做讲座。其中，Jared Lander 和 Ben Reddy 主要负责辅导我们的课后实验，在他们的帮助下，我们才得以在这门快节奏的课上不掉队。他们给我们展示了数据科学的结构。我们的课程覆盖广泛，从线性回归到随机森林算法。我们还认识了很多新的工具：正则表达式、LaTex、SQL、R、Shell 脚本、git、Python。我们掌握了通过 API 和网页抓取获取新数据源的关键技术。

来上这个课的人五花八门，每个人都有需要补充的知识。计算机科学家需要快速学会基本的特征选择理论和使用 R 语言；社会学家需要了解数据库的工作原理，需要知道全局变量和局部变量的区别；金融专业的需要学点道德伦理。大家的学业负担都各不相同。通过慢条斯理地用 R 写循环（虽然最终发现还写错了），或是思考程序如此低效的原因，我们的知识在一点一点增加。图 15-2 即是我们从众多教训中学到的一例。

```
#WARNING THIS TAKES A LONG TIME AND MAKES YOUR COMPUTER GET REALLY HOT
detailed_results <- predict(model,as.matrix(testmatrix),type="raw");
```

图 15-2：从教训中学到的

随着技能的增加，我们能更多的将精力放在如何分析数据上。最终，我们做到了眼中无代码、心中有想法的境界。

这些是仅凭个人能力就能做到的吗？有人能拿下所有的东西吗？

这需要花费数小时不停地和错误做斗争，才能越过学习曲线，这时我们才领略到了数据科学之美。为了按时完成作业，我们需要互相学习，参加本课的同学大都拥有不同的背景知识。

事实上，为了完成作业，找到知识结构上和自己互补的同学甚至是必须的。Rachel 虽然没有要求我们团队协作，但她布置了大量作业，其中不少作业所需要的知识点都比较分散。我们不得不自发组成团队。原来 Rachel 是要让我们知道，数据科学本质上是一门需要合作的学科。开课之初，Rachel 向我们展示了一种类似汽车轮毂状的网络，她让我们绕她围成一圈，辐条将每个人和她连在一起。她希望在课堂上，我们可以建立起新的友谊、提出新的想法、完成新的项目、形成新的联系。

对于这种刚刚萌芽的学科，参与到社区中去变得异常重要。以数据科学为例，社区不仅利于职业生涯发展，而且对于从事日常数据科学相关活动也很重要。如果你不读相关的博客、关注推特上的人，或者不参加行业内的聚会，你如何能知道最新的分布式软件，或者对于某篇著名文章中用到的统计学方法，业内出现了驳斥声音？社区与我们息息相关，在四月份的一次聚会上，Cathy 谈起了 MapReduce，她马上可以就一个问题咨询在场的观众

Nick Avteniev——他是这方面的专家。在社区里，这样的事经常发生。数据科学的知识体系是不断在变化的，而且分散在不同地方。要知道哪些知识是你应该掌握的，唯一方法是看看别人都知道些什么。邀请不同领域的讲师来讲课为我们开启了这一旅程。所有的讲师都乐于回答我们的问题，他们给我们留了他们的电子邮件地址，有些甚至为我们提供了工作机会。

在听过这些专家的课，并且和他们聊过之后，我们有了更多的问题。怎么样在 R 里创建一个时间序列对象？为什么为那个该死的矩阵绘制散点图时不断的出错？随机森林到底是什么鬼东西？我们不仅求助于周围的同学，还同时求助于网上社区，比如 Stack Overflow、谷歌新闻组和有关 R 的博客。我们惊喜的发现，有那么多社区帮助像我们这样的菜鸟数据科学家成长，帮助我们让自己的代码跑起来。我们不仅仅从以前遇到类似问题的人那里得到答案，这些答案早就被发明这些方法的先驱们解答过了。比如 Hadley Wickham、Wes McKinney、Mike Bostock，都为他们写的程序包提供支持。酷毙了！

15.4　殊途同归

世上并不存在柏拉图式的数据科学知识仓库，让你在里面浸淫一下就能掌握相关知识技能。数据科学中所涉猎的各学科都各有优势，其专业词汇各不相同，有时对同样的方法也有不同的解释（正则化参数是先验概率还是仅仅为了平滑曲线？我们该通过原则性理由还是为了要最大化拟合模型选择参数？）没有什么现成的规矩可循，因为规矩还没定出来，这就是为什么交互作用的结构如此重要的原因：你可以制定自己的规矩。你可以自行选择接受谁的影响，像 Gabriel Tarde 说的那样（Bruno Latour、Mark Hansen 都曾援引他的话）：

> 当一个年轻的农民看到日落时，不知道他是选择相信老师说的日心说——这是地球运动的结果，还是选择接受自己的感觉，认为这是太阳运动的结果。这时，这个年轻人心中仿佛有一束射线，经过他的老师，将他和伽利略联系在一起。
>
> —— Gabriel Tarde

选择站在巨人的肩上没有错，但是在爬上巨人的背上之前，或许应该确认一下巨人是否能承受这一重量。在商业上，使用数据卖广告是个热门话题。你或许有权对世界上最好的数据集进行分析，但是如果有人雇用你仅仅是为了找到多卖几双鞋的方法，这样做真的值得吗？

在我们做作业、对答案时，明显发现不同的决定能导致分析结果千差万别。从做出假设到得到最终结果，即使你掌握了其中所有的分析步骤，但由于每一个步骤中仍然存在多种选择，不同选择的组合将是一个庞大的数字。即使如此，那也不是简单地将一个命令的输出传入下一个命令。算法是可裁剪的，选择哪种算法，使用哪些变量更是如此。

来自 Media 6 Degrees（M6D）的 Claudia Perlich 连续在 2003 年、2007 年、2008 年、2009 年赢得 KDD 杯数据挖掘竞赛，现在她是这项赛事的组委会成员。她慷慨地和我们分享了

关于数据科学的得失利弊，以及做决定时采取的不同方式。有次竞赛是预测医院的患者治愈率，她发现患者是按顺序编号的，因此，来自同一个诊所的患者都是连号的。不同医院的条件和患者疾病的严重程度都是不同的，因此，患者的编号成了预测治愈率的一个重要指标。显然，将这种数据漏洞包含在内并不是有意为之，它让整个比赛变成了小菜一碟儿。但是在现实中，它应该用在预测模型中，毕竟，医生和病人选择的诊所是应该被用于预测患者治愈率的。

David Madigan 强调了在数据科学领域做决策时来自道德方面的挑战，通过对制药业的观察研究发现，预测结果也常常差别很大（他向我们展示了阿司匹林的散点图）。他强调除真实数据之外，不能忽略研究者自身的重要性。仅仅调整模型和方法，并且将它们应用到数据上是不够的。

学术界也面临和工业界类似的问题，尽管出于不同原因。在各学科之间，数据科学的差别是如此之大，以致于不能通过单独对某门学科的研究得出数据科学总体的样貌，这些四分五裂的东西究竟是怎么攒在一起的？它们能攒在一起吗？下面这个例子展示了从纯学术的观点是如何量化问题的，这个例子是一道作业题，来自 *The Elements of Statistical Learning*（《统计学习基础》）的“Linear Methods for Regression”（线性回归方法）一章：

> 练习 3.2 已知变量 X 和 Y 的数据，试拟合一个三次多项式回归模型 $f(X) = \sum_{j=0}^{3} \beta_j X^j$，拟合曲线需要达到 95% 的置信带。试考虑如下两种拟合方式：
>
> (1) 对于给定某点 x_0，使线性函数 $\alpha^T\beta = \sum_{j=0}^{3} \beta_j x_0^j$ 在该点的置信区间为 95%；
>
> (2) 对参数 β 求 95% 的置信集合，反过来可计算出 $f(x_0)$ 的置信区间。
>
> 这两种方式有什么不同？哪一个置信带的范围更宽？进行一个小型的模拟实验来比较两种方法的优劣。

这是在机器学习或数据挖掘课上一般会布置的作业。作为初出茅庐的数据科学家，我们现在的第一反应应该是怀疑。在数据分析的过程中，什么时候这样的问题才会出现？在问题出现之前我们都做了些什么？为什么我们只考虑这两个变量，而不是其他变量？我们是如何拿到数据的？谁给的？谁为此买单？为什么要计算 95% 的置信区间？使用其他度量指标效果会不会更好？说真的，谁在乎我们的模型在训练数据上表现得多好？

这对 Hastie 和他的合作者而言并不公平，他们会说如果学生想学如何抓取和组织数据，他们应该找相应的书来看，这是这门课和其他一般入门级课程的鲜明区别。这门课一直在提醒我们，抛开问题的上下文和决策流程，单纯学习数据科学需要的统计学工具是没有意义的。而且，真实数据经常是一团乱麻，处理起来令人头疼；现实中也没人告诉你到底该选择哪种回归模型。但是，仅仅知道这些是不够的，必须通过亲身实践才能理解，纸上得来终觉浅，绝知此事要躬行。

15.5 逢山开路，遇水架桥

用 Michael Driscoll 的话说，我们这群初出茅庐的数据科学家不像那些土木工程师一样，对于正在做的事，我们没有一个全局的视野，蓝图并不是永远存在。数据科学家像冒险家，他们知道自己要寻找什么，他们口袋里有相应的工具，可能是一张地图，或者几个朋友。当他们跋山涉水终于到达城堡时，可能公主并不在那儿，但这都不重要，重要的是他们沿途走来，踩过乌龟，吃过蘑菇，而且依然可以吐火。如果说科学是一排排管道，我们不是管道工，我们是倔强的超级马里奥兄弟！

15.6 作品展示

图 15-3 改进了我们在第 1 章绘制的数据科学家履历表的图形，图 15-4 展示了数据科学在各大学间的流行程度，这些图形都基于 2012 年年底的调查数据。

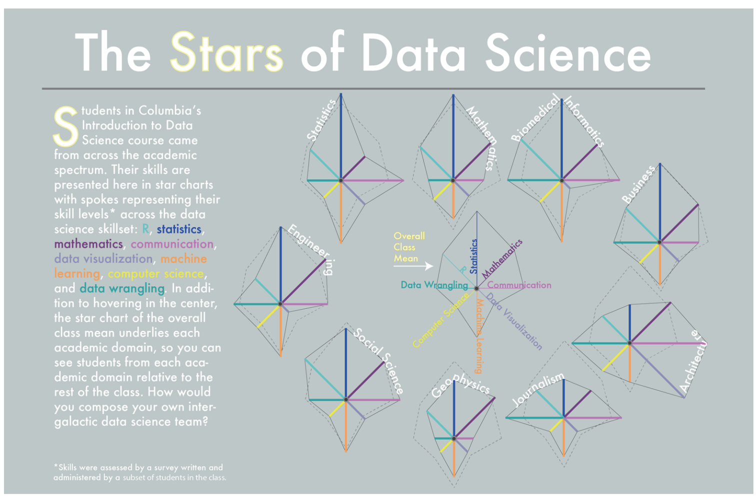


图 15-3：数据科学技能星型图（由 Adam Obeng、Eurry Kim、Christina Gutierrez、Kaz Sakamoto、Vaibhav Bhandari 合作完成）（另见彩插图 15-3）

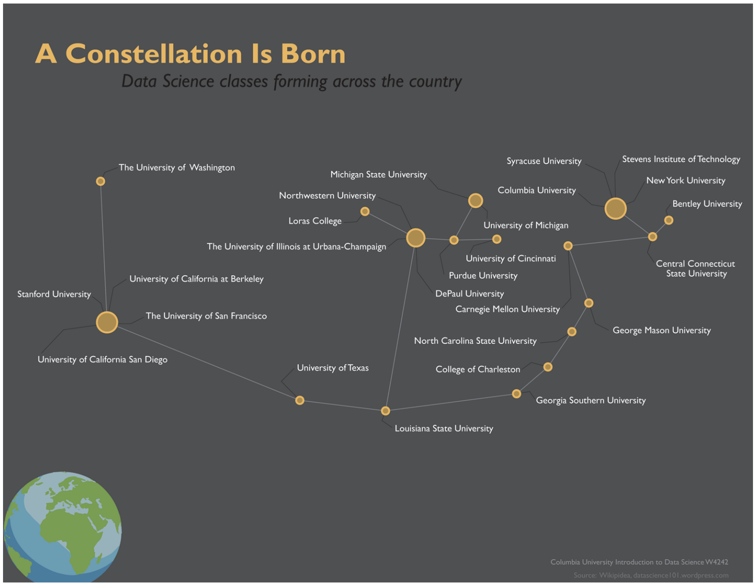


图 15-4：数据科学在各大学间流行程度的星座图（由 Kaz Sakamoto、Eurry Kim、Vaibhav Bhandari 合作完成）（另见彩插图 15-4）

第 16 章

下一代数据科学家、自大狂和职业道德

本章将回顾过去，展望未来，以此对全书做以总结。

16.1　前面都讲了些什么

本书有两个主要目标，一是告诉读者数据科学家是干什么的，二是教会读者一些数据科学家的技能。

我们希望本书已经完成了这两个目标。

针对第一个目标，本书若干章的贡献者为我们带来了大量的关于数据科学家日常工作的一手资料。在本书中，即使没能如我们希望那样全面覆盖数据科学的方方面面，但本书的广度仍然值得我们自豪，这帮助我们实现了第二个目标。

有人可能会写出一本更好的书，下栏里的内容是我们关于这个问题的一点想法。

思维实验：如何教授数据科学

换作是你，你会怎么样去写一本关于数据科学的教材？

首先，数据科学并没有一个确切的定义，也没有标准可循，它更多地流行于报刊和媒

体，但是没有哪家“权威机构”去纠正那些漏洞百出和以讹传讹的错误。其次，数据科学和很多其他学科都有交叉，比如，一本数据科学的教材很容易就会变成另一本机器学习的教材。

如何衡量一本数据科学教材是否成功？如何衡量它的影响力？反之，如何鉴定一本教材是失败的？

我们能把这当成一个数据科学问题吗？如果能使用一个因果模型去回答这个问题，那再好不过了。这需要找到那些和本书读者类似的那些人，但是他们暂未购买本书，然后使用倾向分数配对法进行分析。或者我们可以做个实验，随机挑选一些人，让他们接触不到本书（这有点难，因为人人都可以访问亚马逊网站），而且他们还可以通过其他方法得到。但这都无关紧要，重要的是这比蒙着眼睛猜要强多了。

在业界，一直流传着这样一种说法，数据科学是不可能在大学里或者书本里学会的，必须在工作中边干边学。但这种说法可能不对，这本书可能就是一个证明。学没学会，亲爱的读者，你们说了算。

16.2 什么是数据科学（再问一次）

在书中，我们一次又一次地讨论这个问题，它是本书的主题，是本书的中心问题，都快成口头禅了。

可以简单地通过数据科学家的工作来定义数据科学，在前面讨论数据科学家的知识结构时，我们就这样干过。事实上，在 Rachel 来到哥伦比亚大学开设这门课程之前，她就列了一个数据科学家的工作项目清单，只是她不愿意拿出来给人看，因为这个清单有点乱，而且条目多得有点吓人。这份清单就是后来提出的数据科学家的知识结构的原始材料。在课程结束后，通过与一些人的交谈，她发现他们希望看到这份清单，于是我们就把它列在下面：

- 探索性数据分析；
- 可视化（用在探索性数据分析和汇报中）；
- 数字面板和矩阵；
- 对业务的洞察力；
- 数据驱动决策；
- 数据工程和处理大数据的能力（Mapreduce、Hadoop、Hive、Pig）；
- 收集数据；
- 构建数据管道（日志→ mapreduce →数据文件→同其他数据合并→ mapreduce →清除一些噪声→合并）；
- 开发新产品，而不仅仅停留在对现有产品的使用进行描述上；
- Hack；
- 写专利；

- 数据侦探；
- 预测未来的行为或性能；
- 将发现写成报告、做讲演或发表在学术期刊上；
- 编程（要熟练使用 R、Python、C 和 Java 等语言）；
- 条件概率；
- 优化；
- 算法、统计模型和机器学习；
- 讲故事的能力；
- 会提问题；
- 做调查；
- 搞研究；
- 从数据中做出推测；
- 开发数据产品；
- 找到处理数据的方法，会根据数据规模改变分析策略；
- 一致性检查；
- 对数据的直觉；
- 和领域专家打交道的能力（或者让自己成为一个领域专家）；
- 设计和分析实验；
- 发现数据间的相关性，并且尝试建立潜在的因果关系。

但是现在，我们想再往前走一步，去追求一些更深刻的东西。

在本书的开头，我们将数据科学定义为在科技企业的一组最佳实践。经过了这么长时间的探讨，现在不妨把眼界放宽，不要把数据科学局限在科技企业里，将其他领域也包括进来：神经学、健康状况分析、电子搜索、计算社会学、数字人文研究、基因学、政治等，将所有可用数据解决的问题包括进来。这些问题都可以利用本书讨论的一些最佳实践来解决，只不过有些实践最早在科技企业建立起来而已。数据科学得以同时在工业界和学术界发展，因此，数据科学应用在哪里或哪个领域并不重要，它的关键是通过算法和代码定义了一套从“题域”到“解域”的映射，而数据是关键中的关键。

因此，我们可以这样定义数据科学：数据科学是一些科技公司的最佳实践，但可推广到所有可以用数据解决的问题，有时叫它科学也无妨。即使这样，有时数据科学也只是沦为一场炒作，这是我们要极力避免的，我们更不应该推波助澜地帮助他们炒作。

16.3 谁是下一代的数据科学家

“在我的时代，那些最聪明的人都在思考怎么让用户点击广告……这简直弱爆了。”

—— Jeff Hammerbacher

理想情况下，这一代正在接受训练的数据科学家不应满足于成为技术高手，在一个舒适的城市找一份薪水不菲的工作，当然这些都不错，但我们应该有更高的追求。我们鼓励下一代数据科学家成为提出问题并解决问题的人，深入思考合适的设计和流程，负责任地使用数据，让这个世界变得更好，而不是更坏。让我们在下面几节详细展开这些内容。

16.3.1　成为解决问题的人

让我们先来讨论数据科学家需要具备的技术。下一代数据科学家应努力具备以下技能：编写程序、统计学、机器学习、可视化、沟通的技巧和数学。同时，有一个坚实的编程基础，良好的编程实践，诸如结对编程、代码审查、调试和版本管理，无疑是很有价值的。

什么时候强调探索性数据分析都为时不晚，这点我们在第 2 章讲过，同样的，还有 Will Cukierski 提到的对特征进行选择。Brian Dalessandro 强调了数据科学家如何在无数的模型中做出选择——该选用哪种分类器、特征、损失函数、优选法和评价模型的标准。Huffaker 讨论了特征或矩阵的构成，从日志中变换变量，构建 0-1 变量（比如，重复 5 次同样活动的用户），聚合和计算。虽然这些是数据科学中的关键部分，但经常被认为是微不足道的，因此常常在工作中被忽略了。Dalessandro 将这称为“数据科学的艺术”。

另一个警告：很多人拿到数据就直接使用一个花哨的算法。但是在数据和算法之间，还有很长的一段距离。跑一段代码去预测或分类，当算法收敛时就可以宣告取得了成功，这些都很简单。难的是正确地分析和预测，确保结果是正确和说得通的。

> **下一代数据科学家会怎么做**
> 下一代数据科学家不会试图用复杂但并不奏效的模型来让自己看起来高深莫测。他们花大量的时间整理数据，大概 90% 的时间，尽管没人愿意承认这一点。最后，他们不会陷入对某种工具、方法或某个学术部门的宗教式崇拜中，他们多才多艺，在各学科间切换自如。

16.3.2　培养软技能

很多人都能实现 *k* 最小近邻算法，但很多人都没做好。事实上，大多数人开始做得都不是很好，从哪儿开始不重要，重要的是最终能走多远。养成良好的习惯，以开放的心态持续学习是非常重要的。

我们认为下述一些思想品质有助于解决问题[1]：持之以恒，思考你的大脑是如何思考的，不钻牛角尖，能灵活地思考，永远追求准确度，带有同理心地去思考问题。

注 1：摘自 *Learning and Leading with Habits of Mind*，Arthur L. Costa、Bena Kallick 编（ACSD）。

让我们换个角度来看这个问题，在传统教育体系中，我们关注的是答案。但我们更应该关注，或者至少应该花更大力气去强调的是学生在面对未知问题时该怎么办。我们需要具备能帮助找出答案的素质。

说到这里，你是否想过，为什么人们不知道某件事时，不直接说“不知道”？这种现象，可以被部分解释为“达克效应”，无知比知识更容易招致自信。

基本上，在某件事上不擅长的人，不知道他们在这件事上有多无知，因此容易高估自己，而那些擅长某事的人，却因为了解，反而低估了他们的能力。知识削弱了人们的自信。将这些谨记在心，不要高估，也不要低估自己，确保说到的东西可以做到，和其他数据科学家交谈时不要忘记随时检查自己。

思维实验：如何教授数据科学

换作你，如何设计一门数据科学课程，将重点放在思考习惯，而不是技术的培养上？你如何量化它？又如何评价它？哪些内容学生们可以写在自己的简历上？

16.3.3　成为提问者

人们很容易将模型过拟合，这是人们的天性，每个人都“望子成龙”，很可能你已经在这个模型上工作几个月了，对待它，你会像个慈母或慈父。

人们的另一个天性是低估了坏消息，并且将坏消息归罪于别人。站在父母的角度，自己孩子干的事，或能干的事都不会是坏事。除非是有人使坏，唆使孩子干的。我们该如何克服人类的这种天性？

理想情况下，我们希望数据科学家配得上“科学家”这个称号，他们应该对假设进行检验，不惧挑战，欢迎其他理论进行争鸣。这就是说，我们得挑自己的刺，接受挑战，像个科学家那样设计实验，而不是用花言巧语或以政治为由为自己的模型进行辩护。如果有人说他们能做得更好，那就事先确定好评价标准，让他们去试好了。尽量让事情变得客观。

习惯遵循一份包含关键步骤的标准清单：非要这样做不可吗？如何衡量它？哪种算法适合这个问题，为什么？我该如何评价它？我真的具备做这件事的技术吗？如果没有，我该如何学习这些技术？我可以和谁一起工作？有问题可以问谁？对现实世界有何影响？最后这个可能是最重要的一个问题。

其次，习惯向别人提问。当你遇到一个问题或某个人，需要提出问题时，先假设自己是聪明的，不要认为回答的人比你知道得多或者少。不要试图去证明什么，你的目的是找到答案。像个孩子那样充满好奇心，别怕自己看起来很笨。要求澄清一些符号、术语和流程：数据从哪里来？如何使用这些数据？为什么这些数据能用？我们忽略了哪些数据，这些数

据包含更多特征吗？谁该干什么？如何一起合作？

最后，有一个特别重要的概念要时刻牢记，那就是因果关系和相关关系。不要将二者混为一谈。也就是说，当你看到相关关系时，不要误以为那是因果关系。

下一代数据科学家会怎么做

下一代数据科学家仍然对一切保持怀疑的态度：怀疑模型本身，模型会在什么情况下失败，模型该如何使用，模型又会被如何误用。下一代数据科学家知道他们正在构建模型的影响和后果，他们会思考基于反馈的循环和潜在的在模型之间进行博弈。

16.4 做一个有道德感的数据科学家

数据科学家绝不是一个坐在墙角的书呆子，当你工作时，会有越来越多的伦理问题需要考虑。

现在我们拥有海量的市场和用户行为数据。作为数据科学家，我们不只是机械地使用一些机器学习工具，我们还需要从人文主义角度，去解释和发现数据中的意义，去做出符合道德规范的、基于数据的决策。

要牢记用户行为产生的数据已经构成了数据产品的一部分，反过来数据产品又被用户使用，影响用户的行为。这种例子俯拾皆是：推荐系统、排名算法、好友推荐系统等。这种现象将会在越来越多的行业看到，比如教育、金融、零售、健康等领域。这种基于反馈的循环也会出错，金融风暴就是给我们的一个警示。

数据科学被用来预测未来（Nate Silver，http://slate.me/1g3Di1Y）、预测现在（Hal Varian，http://googleresearch.blogspot.com/2009/04/predicting-present-with-google-trends.html）、探索数据中存在的因果关系（Sinan Aral，http://caossnyc.org/#schedule），这样的例子我们已经讲了很多。

下面我们要讲的是模型和算法不仅用来预测未来，它们还在影响未来。有时，这是我们所期待的，有时，这又是我们想极力避免的。

Emanuel Derman 提出了为金融行业进行建模的“希波克拉底誓言”，这份誓言不仅适用于金融业，也适用于其他行业，让我们以此为起点，介绍一下建模时需要做出的道德考量。

- 时刻谨记这个世界不是由我创造的，它不必符合我的方程式。
- 虽然可以大胆地使用模型做预测，但切不可过分迷信数学。
- 我绝不会牺牲真实换来优雅的数学模型，并且拒绝解释这样做的原因。我也不会在模型的准确度上欺骗我的用户，相反的，我会如实陈述模型的假设条件和模型的局限性。

- 我知道我的工作对于社会和经济影响巨大，其中很多已经超出我的理解能力。

这份誓言并没有将在业界工作时的政治因素考虑进来，但作为一个数据科学家，有时不得不如此。即使对模型心存怀疑，仍然会有人置你的警告于不顾，错误地使用模型。所以“建模者的希波克拉底誓言”的约束力在现实中还是显得有点捉襟见肘，但不可否认，这仍然是一个好的开始。

> **下一代数据科学家会怎么做**
> 下一代数据科学家不会让金钱蒙蔽了双眼，不会将自己的模型用于危害社会的活动。他们寻找机会去解决那些对社会有价值的问题，并且试图了解他们的模型对社会的影响。

最后，有一些使用数据科学服务社会的途径：比如可以作为志愿者参与 DataKind（http://www.datakind.org/）的一些长期项目，这可比那些在周末举行的黑客马拉松有意思多了。

还有让社会变得公正透明的途径：Victoria Stodden 创办的 RunMyCode 网站（http://www.runmycode.org/CompanionSite/），旨在让科研工作开源化和可复制化。

我们想暂且把发言权交给来自哥伦比亚大学历史系的 Matthew Jones 教授，他是历史方面的专家，他也上了我们的数据科学导论课程，他将自己对该课程的一些想法写下来，着重阐述了道德和自大情绪的克服在数据科学中的重要作用。我们把他的文章抄录在此，以飨读者。

数据和自大狂

在 2012 年美国总统大选后，一种幸灾乐祸的情绪在数据科学家以及那些崇拜甚至神化他们的粉丝中迅速蔓延，因传统的专家们预测错误了。计算统计和数据分析彻底击败了传统的预测方式，这些预测一般是基于旧式直觉、长期的新闻业从业经验，或者是依靠日渐式微的华盛顿内部人士关系网。奥巴马团队和其他人基于量化的预测（http://ti.me/GzJlhH）的成功（http://lat.ms/1hkOxki），显而易见（http://ti.me/GzJlhH）地揭示了政治分析领域一个新时代的到来。传统的所谓“专家意见”，现在鲜被提及，而且有人建议应被请下神坛，让位于新出现的数据驱动的政治分析。

这是一段引人入胜的传奇，充分展示了新旧两种知识在面对同一个问题时的冲突。然而，那些真正优秀的数据科学家已经开始反思，完全抛弃现有的领域知识和专家们是相当危险的。

关于起源的故事总会为专业知识的分层增加更多的合理性。数据挖掘领域长期以来有一个广为传颂的故事，尽管有点虚构的成分。故事是这样的：利用一种关联算法

（https://en.wikipedia.org/wiki/Association_rule_learning），研究者意外地发现，在百货商店里购买尿布的男士常常会同时购买啤酒（http://www.itbusiness.ca/news/behind-the-beer-and-diapers-data-mining-legend/136）。其实，市场营销人员凭借他们自学的有限心理学知识和对市场的直觉，早在计算机还被称作“电脑”之前就发现了这一现象。这个故事符合经典的模板，概率论和统计学从欧洲启蒙运动伊始（http://press.princeton.edu/titles/4295.html）就一直在挑战传统知识：保险和养老年金的价格取决于数据，而不是申请者的状况和那些年长的专家们的建议。在将让人喜爱的（或者让人恐怖的）艾普西隆、德耳塔引入真正的分析那本书中（http://goo.gl/v5JhxG），奥古斯丁· 路易· 柯西批评了那些法国大革命中的统计学家：“让我们满怀热情的去发展数学，但不要妄图将其应用于其他领域；我们不要幻想用公式来攻击历史，也不要通过代数理论和微积分来评判道德的高下。”

这些描述与当年兴起的分离主义理念相当契合，而这种理念正是硅谷的自由主义者、信奉熊彼特理论的资本主义者和一些技术期刊的核心主张。虽然可避免政治分析中的权力寻租和其他规则，但二分法错误地分开了技能和知识，而这两者却都是牵引数据科学不断进步的关键。前面的章节主要讲述了培养数据科学家掌握多种技能的各种方法——这将粗浅的二分法观点驳斥得体无完肤，也就是说，数据专家和传统的专家并非毫无交集。本书在教授技术的同时，让那些自大狂变得谦逊，尤其是那些自负算法无敌的人。

奥巴马的数据团队这样解释他们的成功（http://goo.gl/sB8pGH），成功来自严肃地对待自负情绪，构建的技术系统避免了过分估计，选择了将算法作为后台和网络的补充。Haper Reed 向亚特兰大报的作者 Alexis Madrigal 这样解释道：“我认为共和党搞砸了。我知道我们有最好的团队，但是我们不知道它是否能工作。我曾经信誓旦旦这一定能行，我把身家都压在了上面。我们有时间，有资源，我们做了所有能做的，但还是不起作用。总有些事情会发生的。”

有关“领域知识”价值的讨论在数据科学社区里长期呈现两极分化的趋势。无监督学习其实是克服了对人们习惯的社会和科学分析方式的依赖，正如在奥巴马分析团队中看到的那样（http://lat.ms/1hkOxki）。年仅 29 岁的首席分析官 Daniel Wagner 这样说：

> 在竞选活动中找出“足球妈妈”“服务员妈妈”分组游说这样的方式已经过时了。竞选活动现在已经可以精确定位到每个中间选民。郊区的白人妇女？他们都是不一样的。拉丁裔社区差异较大，不同人有不同的兴趣。数据能给你的只是如何区分出这些差异。

人们渐渐弱化了分组，但是将领域知识带入统计学的运动似乎和正规的数据挖掘一样历史悠久。

《华尔街日报》有篇文章（http://on.wsj.com/15LnZno），现在已经不那么有名了。Peggy Noonan 这样形容奥巴马分析团队的工作：”这就像火星人干的。“竞选活动中参与的人很少，作战室里全是些“高科技不会流血”的物种。与此同时罗姆尼一方的广告也类似。

数据科学基于算法但不等同于算法。算法的使用要基于社会学中的“隐性知识”（http://amzn.to/19huR1W）——这是通过实践得来的知识，不易简单总结成规则，或者根本就不可能总结成规则。使用好算法从根本上说也是一项人为活动——一种非算法的东西。

不去警告这些年轻的数据科学家就会带来很多过拟合的危险，在训练集上采用了噪声数据，或者过多学习训练集以致不能将结论泛化。避免过拟合需要仔细考虑使用的算法。算法是需要我们投入更多思考的工具。Peter Huber 在 1977 年这样解释道：“我认为，问题不是要用机器智能取代人类的聪明才智，而是使用所有计算机科学、人工智能提供的工具帮助人类发挥自己的聪明才智，尤其是即兴使用各种搜索工具、记录分析的进展。”“即兴”一词恰好指出了要掌握工具的使用、依据上下文推理、避免死记硬背。自大狂使用算法时，务必要深入理解算法，对具体的实现要了如指掌。

Noonan 提供的奥巴马竞选团队的招聘广告上（http://goo.gl/3KuIuj），明显体现出对现有模型优点和缺点的思考：

- 开发和实现统计 / 预测 / 机器学习模型支持竞选活动、数字媒体、付费媒体和募捐竞选资金；
- 评价以前模型的性能，决定是否需要更新；
- 设计和执行实验，验证模型的适用性和有效性。

自动化的模型唾手可得，并不是这里的重点：重点是批评和评价。只有熟悉这一领域的火星人才能干这个：各种各样的数据太重要了，千万不可放过。

怎么样学会随机应变？换句话说，什么模型是教育数据科学家的最佳选择？能够使用算法和大数据来即兴发挥的能力需要持续不断地抽丝剥茧，清理组织混乱、不完整、很可能是没有结论的数据。训练需要的最佳模型不是通过这种狭隘的职业教育，而是回归人文艺术的本质。

几个世纪以来，艺术，比如数学和音乐都被归为人文学科的范畴，因为它们不能被自动化、机器化、重复化和惯常化。人文学科让人得到自由，这种自由反映在他们使用的工具、行为和习惯上。人们不被工具所奴役，他们可以自由地选择使用或不使用工具。算法也是如此，称职的数据科学家不必循规蹈矩，任何技术相关的宿命论都不适用。数据科学家从不将使用一项技术的可能性和必要性混为一谈。面对数据有无数可能，但是只有很小的一部分合乎道德（并且有趣）的问题需要我们花力气去解决。

—— Matthew Jones

16.5 对于职业生涯的建议

对于雄心勃勃的下一代据科学家，尤其那些已经阅读至这里的读者朋友们，我们是从来不吝惜自己的建议的。

我们已经习惯回答这类问题了，毕竟，很多人都曾问过我们他们是否应该成为一名数据科学家。相比直接给出答案，我们经常会先问他们两个问题。

1. **你选择什么样的生活**

   要回答这个问题，你需要清楚自己看重什么。你也许看重金钱，你需要足够的钱过上你理想的生活，甚至想要更多的钱。这样很多很酷、但是没人愿意掏钱的项目就被排除在你的选择之外了（但千万不要因此放弃为这类项目筹款）。你也许看重和爱人或者朋友共度美好时光，那么你就不应该选择去一些初创公司工作，在那里的人一天工作十二小时，而且经常睡在办公桌底下。我没骗你，这样的地方的确存在。

   或者你追求的是为世界干点有益的事，同时实现自我价值并体现智慧价值。一定要根据个人情况去衡量这些选择，它们是截然不同的选择。

   你的目标是什么？你想追求什么？你想成名吗？你想受人尊敬，并且学有所长吗？或许你的最佳选择是上述几种情况的某种组合，那么现在，你清楚地知道你的选择了吗？

2. **你有哪些局限**

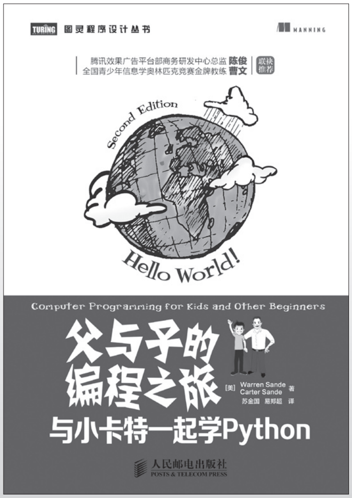
   有很多外在因素，或许是你无法左右的，比如你无法选择和家人住在哪里。还有金钱和时间上的限制，你是否需要研究公司的年假、产假和陪产假政策。向用人单位推销自己难度几何？不要被逼入绝境，想想如何展示自己积极的一面：学历、优缺点、能改变的和不能改变的。

基于你看重的东西和那些局限因素，有很多方案可供选择。在我们看来，工作没有好坏，只有合适不合适。不同的人想从工作中获得的东西是不一样的。

一方面，你决定要做的任何事都不是绝对的，你随时可以改变当初的选择，人们不都在换工作吗？因此不要太过担心。另一方面，人生苦短，不要停滞不前，永远向着正确的方向前进，永远追寻那些内心真正在乎的东西。

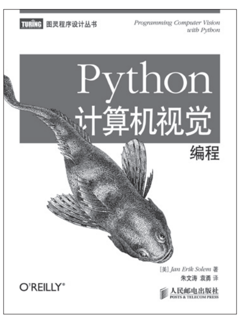
最后，如果你认为你的想法和思考方式和周围人不同，那么请相信自己，去探索它，你有可能大有作为。

图灵最新重点图书

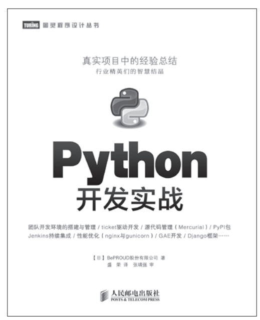


- 程序员爸爸的第一本亲子互动编程书
- 腾讯效果广告平台部商务研发中心总监陈俊
  全国青少年信息学奥林匹克竞赛金牌教练曹文
  联袂推荐
- 内容经过教育专家的评审，经过孩子的亲身检验，并得到了家长的认可

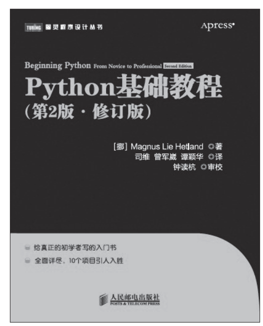
父与子的编程之旅
书号：978-7-115-36717-4
作者：Warren Sande Carter Sande
定价：69.00 元



Python 计算机视觉编程
书号：978-7-115-35232-3
作者：Jan Erik Solem
定价：69.00 元



Python 开发实战
书号：978-7-115-32089-6
作者：BePROUD 股份有限公司
定价：79.00 元



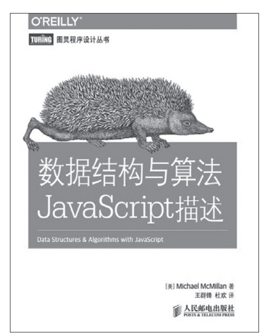
Python 基础教程（第 2 版·修订版）
书号：978-7-115-35352-8
作者：Magnus Lie Hetland
定价：79.00 元



程序员必读之软件架构
书号：978-7-115-37107-2
作者：Simon Brown
定价：49.00 元



AngularJS 权威教程
书号：978-7-115-36647-4
作者：Ari Lerner
定价：99.00 元



数据结构与算法 JavaScript 描述
书号：978-7-115-36339-8
作者：Michael McMillan
定价：49.00 元

读者评论

“这本书告诉我们什么是数据科学。”

“本书是进入数据科学领域的入门指南，它会告诉你干这一行哪些技能是必备的！”

“这本书既严谨，又非常通俗易懂。各种概念的讲解都提供了真实案例辅助理解。”

“本书汇集了行业翘楚的大量洞见。它不仅能让你全面把握这个新兴的领域，来自一线的实战经验也能让你迅速站在行业的前沿。”

O'REILLY®

数据科学实战

大数据时代，人们越来越意识到数据在工作和生活中的重要性，数据科学家应运而生。面对媒体天花乱坠的炒作，怎么才能拨云见日，真正掌握这门跨学科利用数据的学问呢？这本脱胎于常春藤名校哥伦比亚大学“数据科学导论”课程的实战手册能够给你一个满意的回答。

本书作者Rachel Schutt曾在谷歌研究院工作多年，现为美国新闻集团数据科学高级副总裁。她在哥伦比亚大学任教期间，广泛邀请了谷歌、微软、eBay及一些创业公司的数据科学家为学生授课，打破了所谓大学里教不出数据科学家的神话。这些讲座涵盖了上述公司及业界使用的最新算法、方法和模型。本书就是在这些一手资料基础上汇编而成的，它不仅可供不具备相关领域知识的初学者真正了解数据科学，而且也是熟悉线性代数、概率论、统计学、机器学习等主题的人士开阔视野、提升实战技能的优秀指南。

本书内容：

- **统计推断、探索性数据分析（EDA）及数据科学工作流程**
- **算法**
- **垃圾邮件过滤、朴素贝叶斯和数据清理**
- **逻辑回归**
- **金融建模**
- **推荐引擎和因果关系**
- **数据可视化**
- **社交网络与数据新闻**
- **数据工程、MapReduce、Pregel和Hadoop**

Rachel Schutt

美国新闻集团旗下数据科学部门高级副总裁、哥伦比亚大学统计系兼职教授、约翰逊实验室高级研究科学家，同时也是哥伦比亚大学数据科学及工程研究所教育委员会的发起人之一。她曾在谷歌研究院工作数年，负责设计算法原型并通过建模理解用户行为。

Cathy O'Neil

约翰逊实验室高级数据科学家、哈佛大学数学博士、麻省理工学院数学系博士后、巴纳德学院教授，曾发表过大量算术代数几何方面的论文。他曾在著名的全球投资管理公司D.E. Shaw担任对冲基金金融师，后加入专门评估银行和对冲基金风险的软件公司RiskMetrics，个人博客：mathbabe.org。

DATABASES / DATA

封面设计：Karen Montgomery 张健

图灵社区：iTuring.cn
热线：(010)51095186转600

分类建议 计算机/程序设计/数据分析

人民邮电出版社网址：www.ptpress.com.cn

O'Reilly Media, Inc.授权人民邮电出版社出版

ISBN 978-7-115-38349-5

ISBN 978-7-115-38349-5
定价：79.00元

看完了